



Les logiciels CFAO de l'emballage créatif

Guide de formation

PicXDK®

Version 3
Révision 0.2

Microsoft®
CERTIFIED
Partner

PICADOR

10 rue Rossini, 75009 PARIS – tél : +33 1 42 46 77 00 – fax : +33 1 42 46 24 06
www.picador.fr info@picador.fr

Table des matières

<i>AVERTISSEMENT</i>	6
A lire avant toute utilisation.	6
<i>Bienvenue</i>	7
Avant propos	7
La documentation de la technologie XDK®	7
Les nouveaux modules de PICADOR® basés sur la technologie XDK®	8
La Technologie XDK®	9
Support technique	10
Se préparer à un Support technique	10
Pour obtenir le support technique	10
<i>Formation 1</i>	11
Créer un projet VB	11
Ajouter le composant ParamX	12
Insertion d'un bouton de commande	15
Utilisation du contrôle ParamX	16
Exécution du programme	17
<i>Formation 2</i>	18
Saisie des paramètres	18
Création des variables paramètres	19
Exécution du programme	20
<i>Formation 3</i>	21
Modifier les entités créées (Arrondi)	21
Symétrie	22
<i>Formation 4</i>	24
Amélioration de l'interface	24
Bouton Quitter	24
Modification Immédiate (Evénements)	24
Contrôle de la saisie	25
<i>Les Assistants</i>	26
Introduction	26
Installation des Assistants	26

Utilisation des Assistants	29
Liste des Propriétés	35
L' Explorateur d'objets	35
Compte	36
CompteQuestions	36
Couleur	36
Dimension	36
Direction	36
DrawLabelNum	36
fichier	36
Grille	37
Groupe	37
LabelQuestion	37
Niveau	37
Origine	37
Outilvisu	37
Recadrer	37
Librairie des Fonctions Membres	38
AjoutArc	38
AjoutCoteAngle	38
AjoutCoteDiametre	38
AjoutCoteDistance2Pts	39
AjoutCoteRayonArc	39
AjoutFormatCarton	39
AjoutFormatCartonCentre	39
AjoutPose	40
AjoutProfilFenetre	40
AjoutProfilPt	40
AjoutQuestion	40
AjoutSeg	41
AjoutSeg2Pts	41
AjoutTexte	41
AjoutSeg	41
ArcCos	42
ArcSin	42
Arrondi	42
CalcLgFilets	42

CalcRectExInscrit	43
CalculSurfaceProfil	43
DebutParam	43
DlgFiltres	43
DlgMiseEnPage	43
DlgStylos	44
Effacer	44
Filtrer	44
FinParam	44
GetAng_o	44
GetCode	45
GetDeport	45
GetDim	45
GetDir	45
GetGroupe	46
GetNiveau	46
GetRGB	46
GetStylo	46
GetTexte	46
GetX	47
GetY	47
Imprimer	47
ImprimerDirect	47
InsererModFT	48
nomModFT	48
Ouvrir_Avec	48
PtsInterEntites	48
Recuperer	49
Refresh	49
ReIndex	49
SauverSous	49
SetAng_o	49
SetDeport	50
SetDim	50
SetDir	50
SetGroupe	50
SetNiveau	51

SetStylo	51
SetTexte	51
SetX	51
SetY	52
SymParAxe	52
Tranf2d	52
ZoomArrière	52
ZoomAvant	53
ZoomPan	53

AVERTISSEMENT

A lire avant toute utilisation.

- 1- LES INFORMATIONS CONTENUES DANS CE DOCUMENT PEUVENT FAIRE L'OBJET DE MODIFICATIONS SANS PREAVIS.
- 2- CE DOCUMENT EST REMIS AU LECTEUR DANS LE SEUL BUT DE FACILITER LA CONNAISSANCE DU SYSTEME PICADOR® , DONT IL A ACQUIS LE DROIT D'UTILISATION.
- 3- CALTEC DECLINE TOUTE RESPONSABILITE POUR TOUT DOMMAGE POUVANT RESULTER DES INFORMATIONS CONTENUES DANS CE DOCUMENT.
- 4- L'ATTENTION DU LECTEUR EST ATTIRE SUR LE FAIT QU'IL LUI INTERDIT DE DIVULGUER, OU DE FACILITER LA DIVULGATION DE CE DOCUMENT, DE COPIER OU DE REPRODUIRE TOUT OU UNE PARTIE DU DOCUMENT, PAR QUEL QUE MOYEN OU SOUS QUELLE QUE FORME QUE CE SOIT, DE LE TRADUIRE DANS TOUT AUTRE LANGAGE, SANS ACCORD EXPRESS DE CALTEC.

Bienvenue

Avant propos

Bienvenue dans les solutions PICADOR® pour MS-Windows. Ces solutions de **CAO/CFAO** qui vous apporte toute la puissance et l'ergonomie de l'interface graphique la plus utilisée. Disponible sur toutes les plates-formes MS-Windows 32 bits (W95, W98, NT) , vous utiliserez toutes les ressources disponibles sur l'ordinateur personnel ou sur le réseau si celui-ci est présent.

La documentation de la technologie XDK®

Ce guide a été rédigé dans un souci de simplicité et de précision au niveau des informations présentées. Pour chaque fonction, vous trouverez une description ainsi que le déroulement étape par étape de leur mise en œuvre.

Quand cela l'exigeait, nous avons pris soin d'illustrer d'exemples les détails des fonctionnalités de la technologie XDK®

Les nouveaux modules de PICADOR® basés sur la technologie XDK®

Les solutions PICADOR Windows ont été largement engagées depuis quelques années dans l'environnement objet vers lequel tend toute l'industrie informatique. Les langages (C++, J++,...), les systèmes d'exploitation (WNT5, W2000), Internet, les applications mais surtout l'échange et l'intégration des « objets » entre applications sont tous orientés vers cette technologie objet qui modifiera de plus en plus notre approche et nos choix de solutions informatiques.

L'avance prise par PICADOR dans ce domaine (PicGEOM® est entièrement réécrit en C++ depuis 1996, il exploite la librairie de classes Microsoft MFC) nous permet de faire un pas décisif vers ce nouvel environnement logiciel et système en proposant sa technologie XDK.

La Technologie XDK®

Depuis plusieurs années, les systèmes d'information se sont généralisés, multipliant ainsi les formats des fichiers de données. Paradoxalement, peu de solutions ont été apportées afin de faciliter l'échange de données. On ne trouve le plus souvent qu'une simple interface !

Conscient de cette problématique, CALTEC a depuis longtemps développé une technologie capable de répondre à cette attente. Il s'agit de la technologie **IAC** (Interfacer, **A**fficher, **C**onvertir). Celle-ci fournit des outils de communication et de liaison de données afin d'obtenir un système d'information homogène. L'objectif recherché est de dialoguer et d'échanger des données de toute provenance et vers toute destination.

Déjà plusieurs domaines ont bénéficié de cette technologie, notamment le couplage **CAO/CALCUL** (passage d'éléments géométriques vers des éléments finis, ...) ou bien encore l'intégration au sein d'un système de **GPAO**.

Aujourd'hui, avec la technologie **XDK®**, Caltec vous permet d'aller plus loin et d'intégrer des composants objets dans les applications capable de les recevoir et de communiquer avec.

Pour cela, nous avons développé un **serveur OLE automation PicSvOLE**, permettant de réaliser des composants objets indépendants (**contrôles ActiveX**).

Dans les solutions Picador, deux contrôles sont actuellement proposés : **PicViewX** et **ParamX**.

L'un permet visualisation des documents Picador, l'autre permet de programmer, de modifier ou de créer ces documents.

Ils peuvent être utilisés dans des applications telle que Excel®, Word®, Access®, être programmé avec des outils simple comme Visual Basic, être implémentés dans des pages Web, ...

Le but à atteindre restant évidemment le même, amélioré le système d'information, avec plus de productivité de communications et d'échanges au travers de solutions intégrées.

Support technique

Se préparer à un Support technique

Si vous avez besoin d'aide, contactez le support technique de CALTEC. Avant de téléphoner, placez vous devant votre ordinateur avec votre dessin à l'écran et le guide de l'utilisateur de PICADOR à portés de main. Soyez prêt a fournir les informations suivantes :

- La formulation exacte des messages qui sont apparus sur votre écran lors du problème.
- Une description de ce qui s'est passé et de ce que vous faisiez à ce moment-là.
- Ce que vous avez tenté de faire pour résoudre le problème.

Pour obtenir le support technique

Pour obtenir le support technique, vous pouvez vous adresser aux numéros suivants:

Téléphone : 01 55 47 02 55

Fax : 01 47 51 20 78

Courrier électronique : picador@caltec.fr

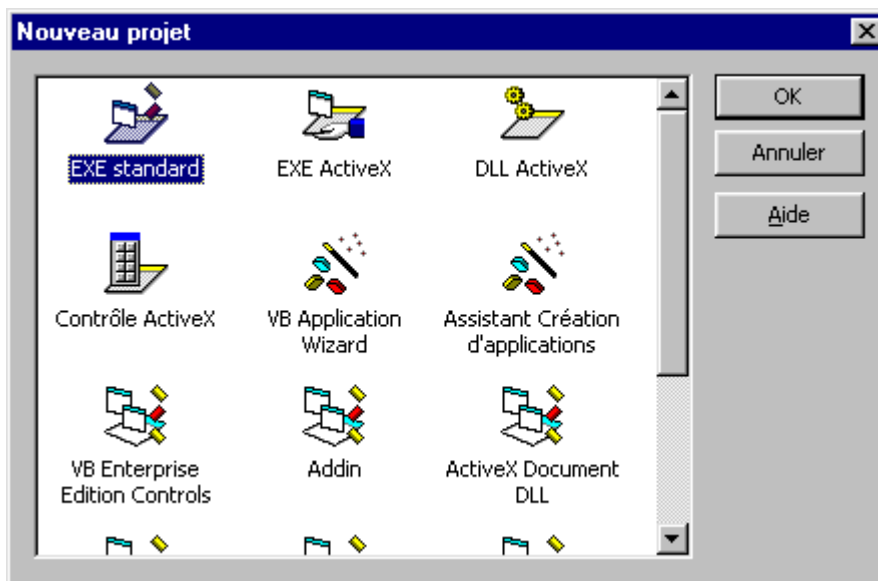
[prenom.nom]@caltec.fr

Formation 1

Créer un projet VB

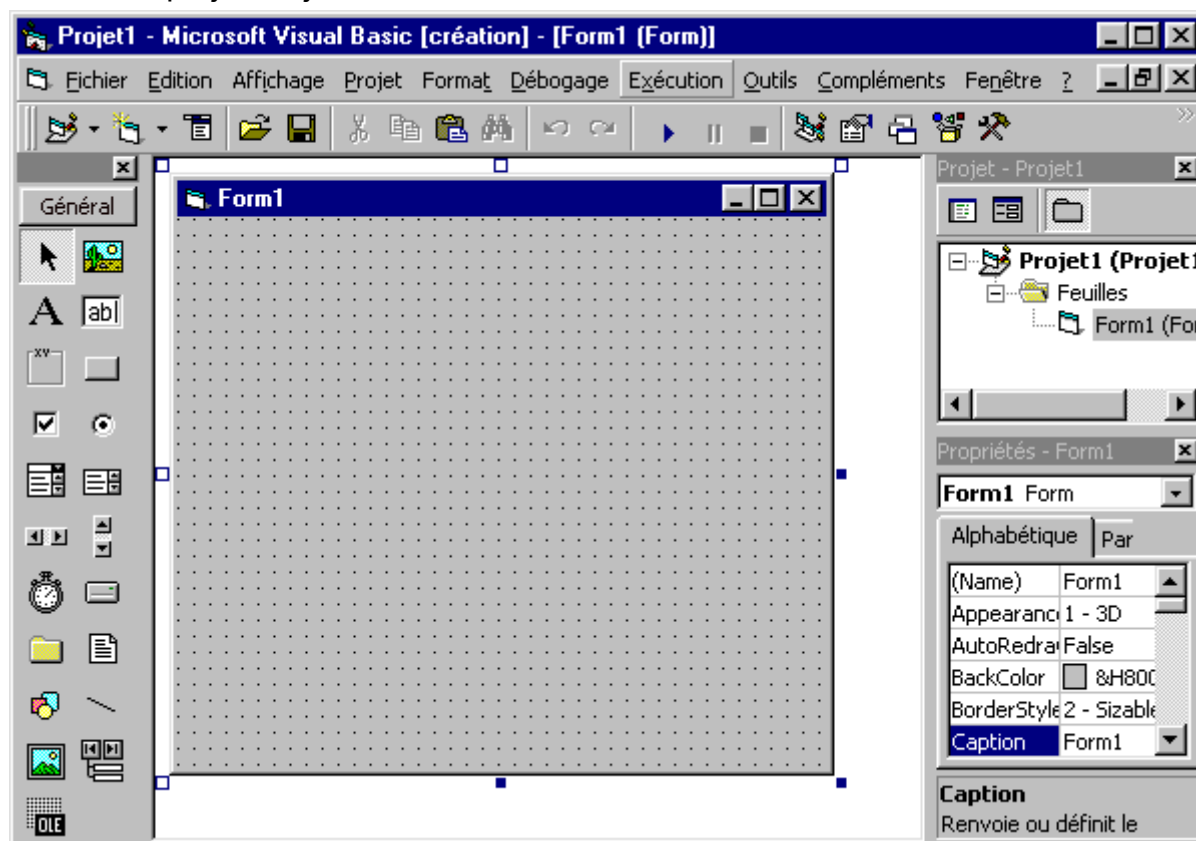
Le but de ce premier exercice est de réaliser avec Visual Basic le paramétrage simple d'un rectangle.

Après avoir lancer Visual Basic, choisir dans le menu "*Fichier*" la rubrique "*Nouveau Projet*".



Choisir "EXE Standard".

Le projet *Projet1* est alors créé avec une Forme nommée *Form1*.

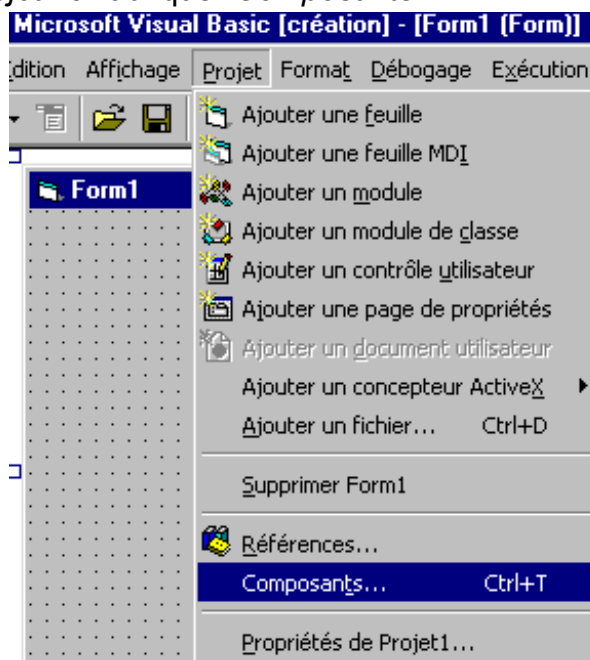


La barre d'outil principale est composée des principaux contrôles permettant de réaliser l'interface utilisateur.

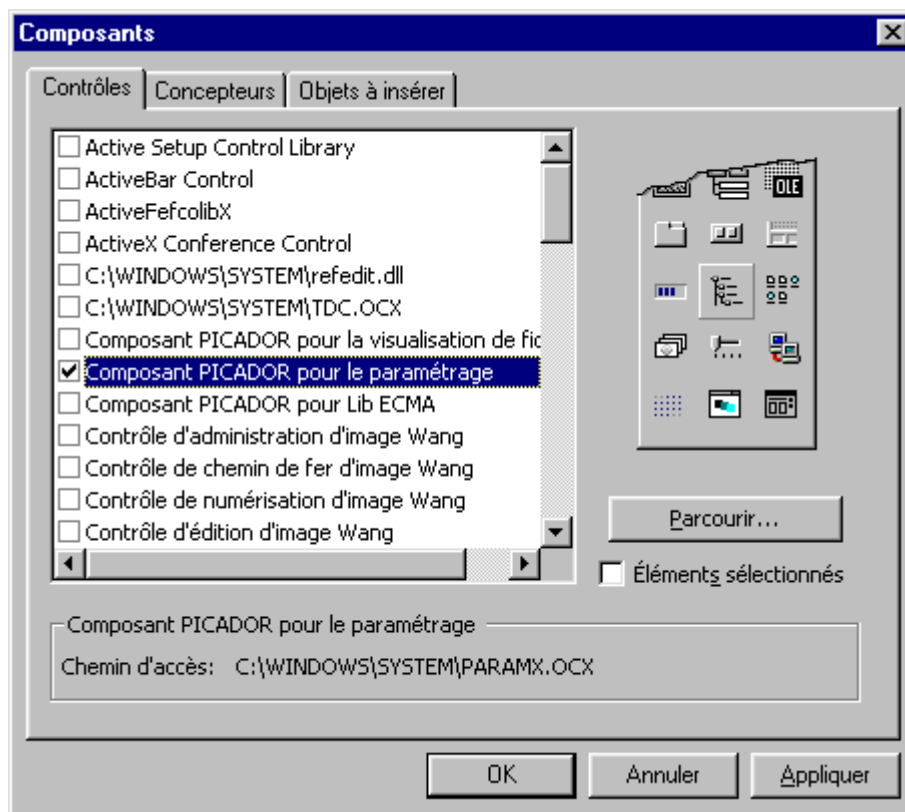
Pour pouvoir réaliser notre paramétrage, nous allons ajouter un composant spécialisé.

Ajouter le composant ParamX

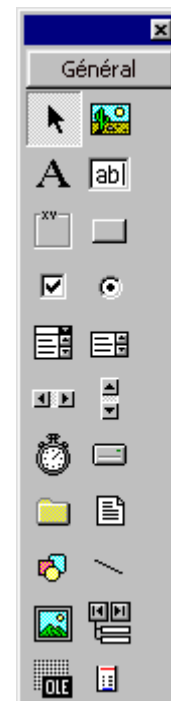
Choisir dans le menu "*Projet*" la rubrique "*Composants*"



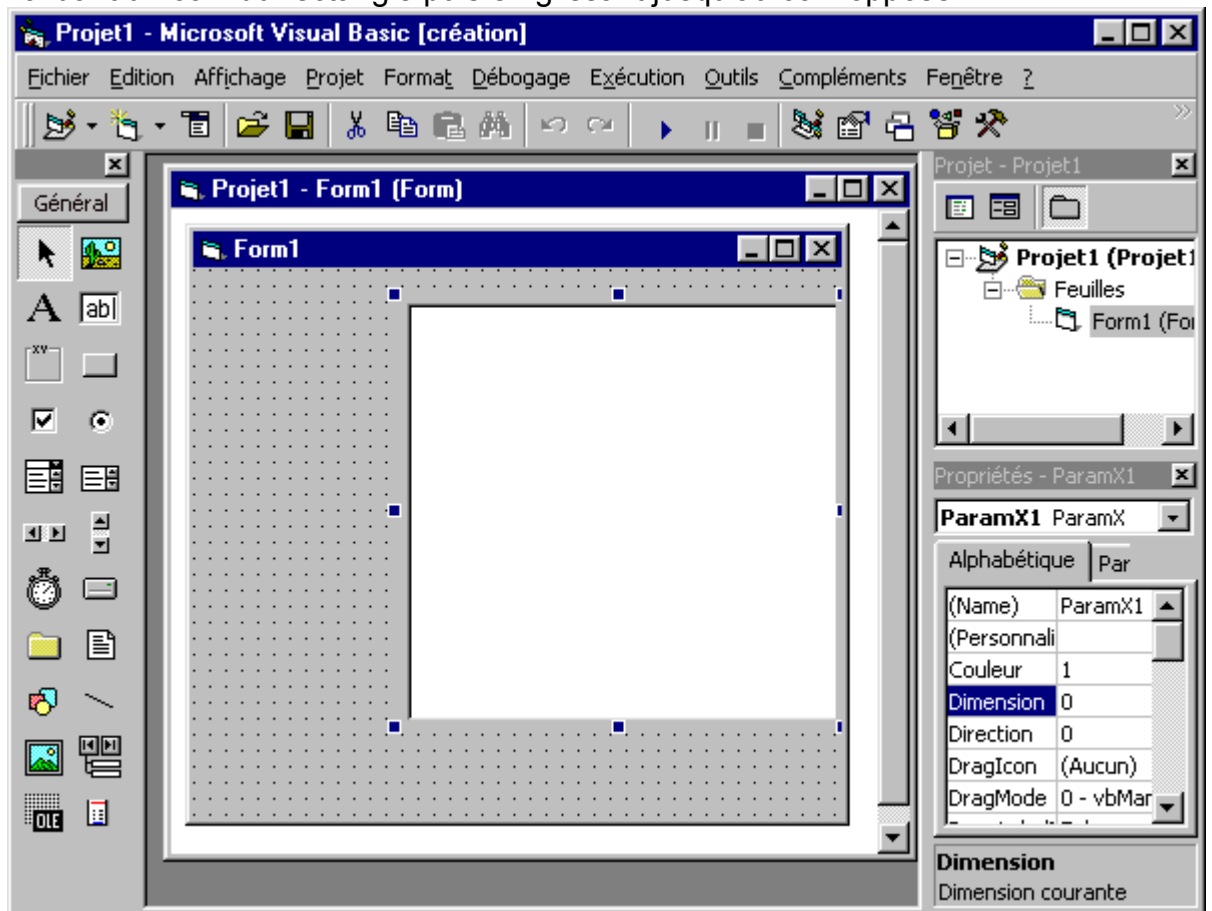
Choisir le composant PICADOR pour le paramétrage.



Un nouveau contrôle est ajouté a la barre d'outil principale.



Sélectionner ce contrôle *ParamX* et le positionner sur la forme *Form1* en validant un coin du rectangle puis en glissant jusqu'au coin opposé.



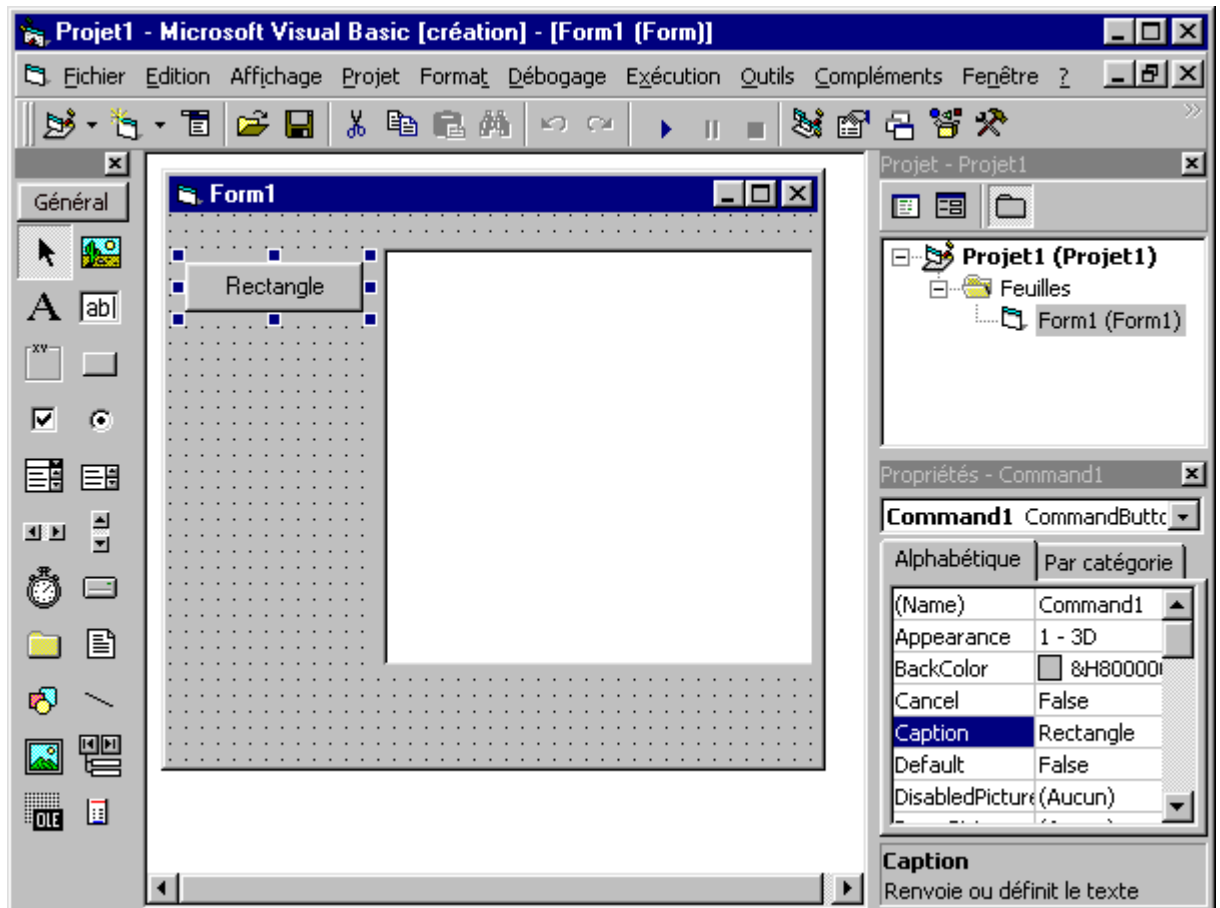
Insertion d'un bouton de commande

Choisir dans la barre d'outil principale le contrôle *CommandButton*



Positionner le bouton en tirant sur un rectangle dans la forme *Form1*.

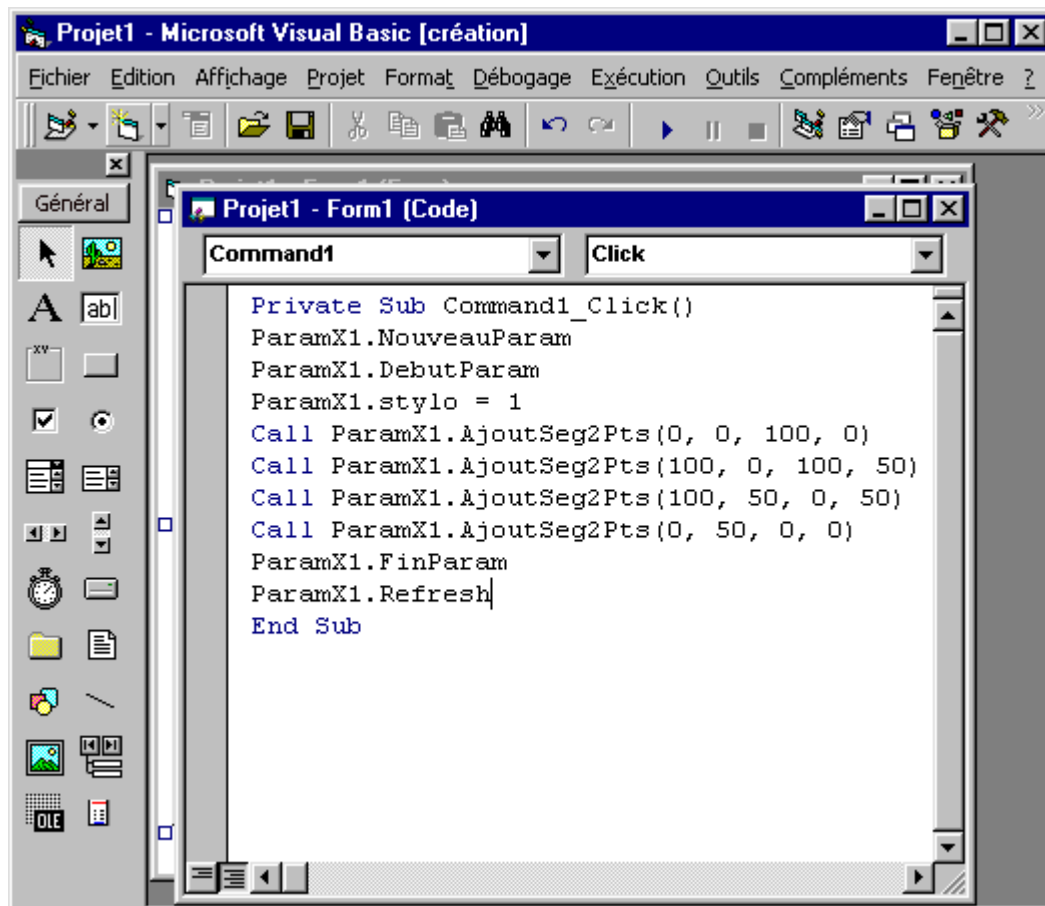
Modifier la propriété *Caption* pour indiquer le nom de commande associée au bouton (par ex. *Rectangle*)



Utilisation du contrôle ParamX

Double Cliquer sur le bouton *Command1* (Rectangle).

Puis taper les commandes suivantes:

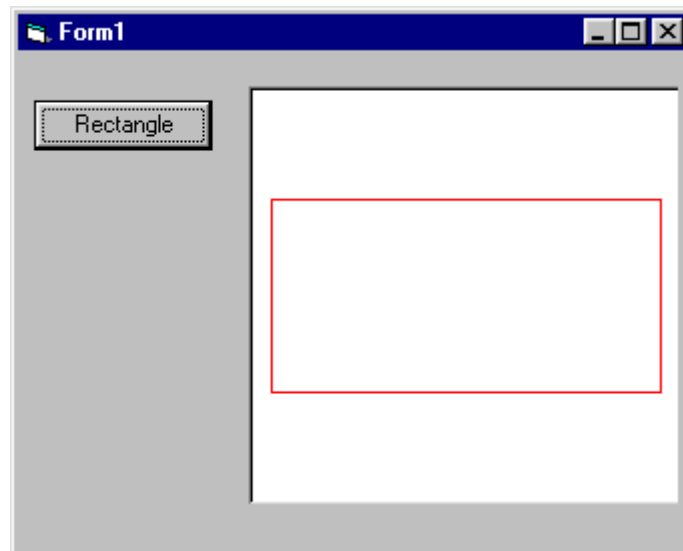


ParamX1.NouveauParam	Ouverture d'un nouveau document
ParamX1.DebutParam	Début de la programmation du document
ParamX1.stylo = 1	Stylo par défaut des entité créées
Call ParamX1.AjoutSeg2Pts(0, 0, 100, 0)	Ajouter un segment en définissant ses 2 extrémités
Call ParamX1.AjoutSeg2Pts(100, 0, 100, 50)	"
Call ParamX1.AjoutSeg2Pts(100, 50, 0, 50)	"
Call ParamX1.AjoutSeg2Pts(0, 50, 0, 0)	"
ParamX1.FinParam	Fin de la programmation du document
ParamX1.Refresh	Rafraîchir le contrôle (Affichage)

Exécution du programme

Lancer l'exécution du programme réalisé en cliquant sur l'icône  (F5).


Valider le bouton *Rectangle* : un rectangle de 100x50 s'affiche dans le contrôle *Paramx*.

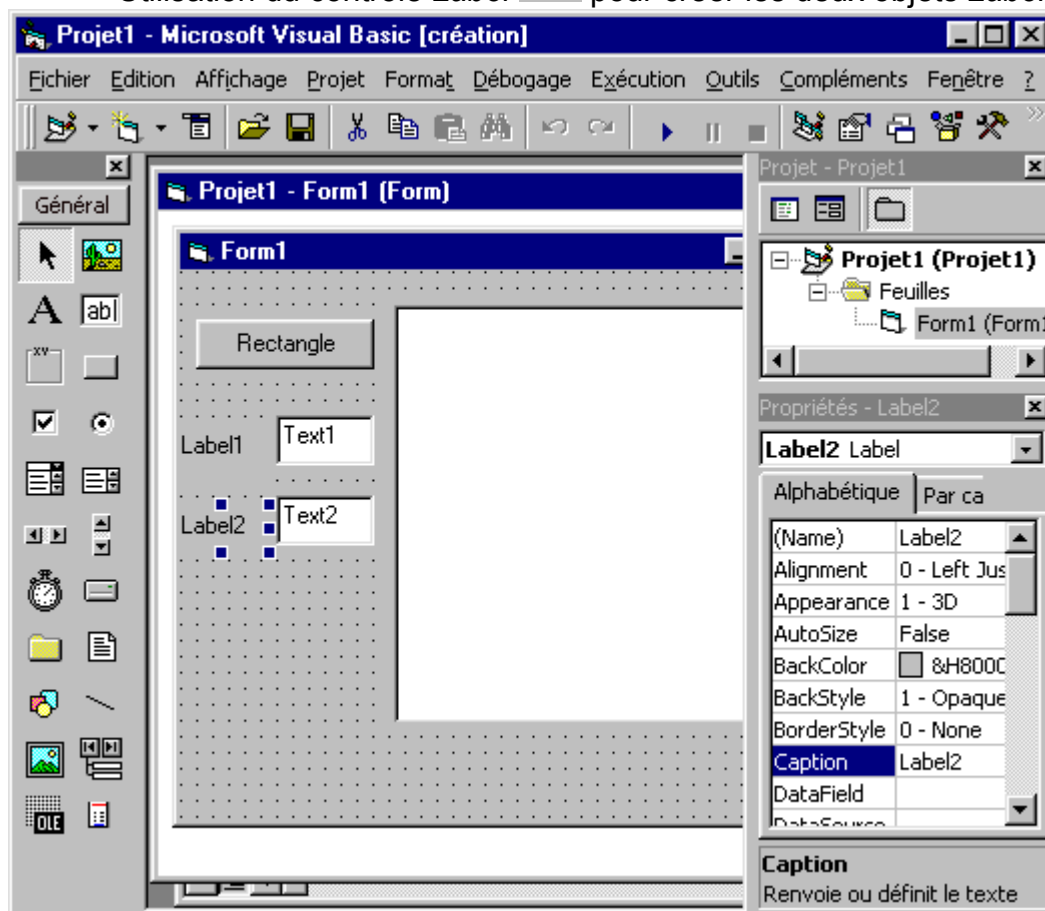


Formation 2

Saisie des paramètres

Utilisation du contrôle *TextBox*  pour créer deux objets *Text1* et *Text2*.

Utilisation du contrôle *Label*  pour créer les deux objets *Label1* et *Label2*.



Modifier la propriété *Caption* de *Label1* et *Label2* pour attribuer les valeurs: *Longueur* et *Largeur*.

Modifier la propriété *Text* de *Text1* et *Text2* pour attribuer les valeurs: 200 et 100.

L'interface utilisateur est prête pour la saisie des 2 paramètres d'un rectangle.

Création des variables paramètres

Modifier le fonctionnement du bouton de commande *Command1* en ajoutant et modifiant les lignes suivantes.

```

Private Sub Command1_Click()
    Dim largeur As Single
    Dim longueur As Single
    largeur = Val(Text1.Text)
    longueur = Val(Text2.Text)
    ParamX1.NouveauParam
    ParamX1.DebutParam
    ParamX1.stylo = 1
    Call ParamX1.AjoutSeg2Pts(0, 0, longueur, 0)
    Call ParamX1.AjoutSeg2Pts(longueur, 0, longueur, largeur)
    Call ParamX1.AjoutSeg2Pts(longueur, largeur, 0, largeur)
    Call ParamX1.AjoutSeg2Pts(0, largeur, 0, 0)
    ParamX1.FinParam
    ParamX1.Refresh
End Sub

```

Dim largeur As Single	Variable (single = réel en simple précision)
Dim longueur As Single	"
largeur = Val(Text1.Text)	Conversion du texte du contrôle par sa valeur numérique
longueur = Val(Text2.Text)	"
....	
Call ParamX1.AjoutSeg2Pts(0, 0, longueur, 0)	Modification des valeurs fixes par les variables.
Call ParamX1.AjoutSeg2Pts(longueur, 0, longueur, largeur)	"
Call ParamX1.AjoutSeg2Pts(longueur, largeur, 0, largeur)	"
Call ParamX1.AjoutSeg2Pts(0, largeur, 0, 0)	"

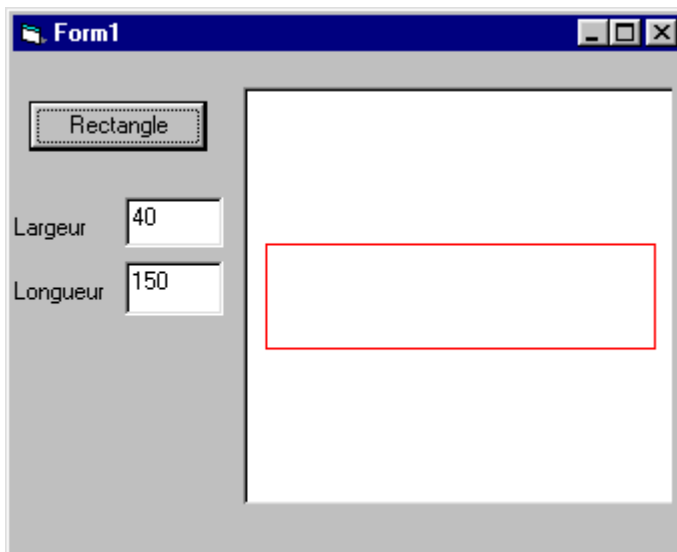
Exécution du programme

Lancer l'exécution du programme réalisé en cliquant sur l'icône  (F5).

Valider le bouton *Rectangle* : un rectangle de 100x50 s'affiche dans le contrôle *Paramx*.

Modifier les valeurs de *Largeur* et *Longueur* puis cliquer sur le bouton *Rectangle*.

Le rectangle ainsi défini s'affiche dans le contrôle *ParamX*.



Formation 3

Modifier les entités créées (Arrondi)

Pour modifier des entités il faut utiliser l'index retourné par les fonctions de création pour pouvoir les désigner .

Exemple : création d'un arrondi entre 2 segments.

Dans la fonction *Command1_Click*, ajouter et modifier les lignes suite

....	
Dim index1 As Long	Variable index1
Dim index2 As Long	"
....	
Call ParamX1.AjoutSeg2Pts(longueur, 0, longueur, largeur)	
index1 = ParamX1.AjoutSeg2Pts(longueur, largeur, 0, largeur)	Repéré le 1 ^{er} segment
index2 = ParamX1.AjoutSeg2Pts(0, largeur, 0, 0)	Repéré le 2 ^{ème} segment
Call ParamX1.Arrondi(index1, index2, largeur / 3)	Créer l'arrondi entre les 2 segments d'un rayon largeur/3
....	

```


Private Sub Command1_Click()
    Dim largeur As Single
    Dim longueur As Single
    Dim index1 As Long
    Dim index2 As Long
    largeur = Val(Text1.Text)
    longueur = Val(Text2.Text)
    ParamX1.NouveauParam
    ParamX1.DebutParam
    ParamX1.stylo = 1
    Call ParamX1.AjoutSeg2Pts(0, 0, longueur, 0)
    Call ParamX1.AjoutSeg2Pts(longueur, 0, longueur, largeur)
    index1 = ParamX1.AjoutSeg2Pts(longueur, largeur, 0, largeur)
    index2 = ParamX1.AjoutSeg2Pts(0, largeur, 0, 0)
    Call ParamX1.Arrondi(index1, index2, largeur / 3)
    ParamX1.FinParam
    ParamX1.Refresh
End Sub

```

Symétrie

Nous nous proposons de réaliser la symétrie de l'ensemble du dessin par rapport à un axe vertical.

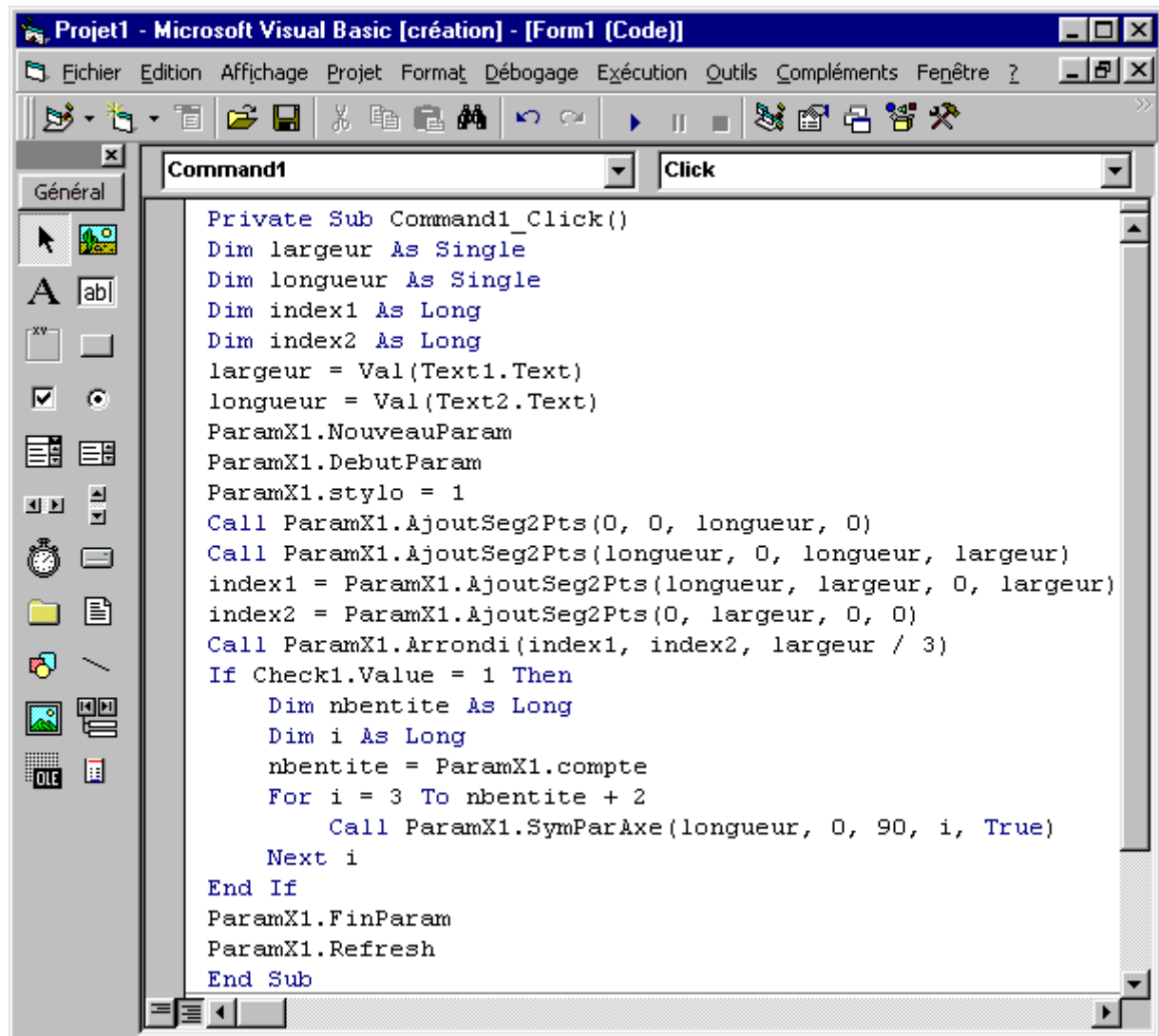
Pour cela, on va réaliser un bouton à cocher pour choisir si l'on veut réaliser la symétrie.

Dans la boîte à outils principale, choisir le contrôle *CheckBox*  puis le positionner sur la forme *Form1*.

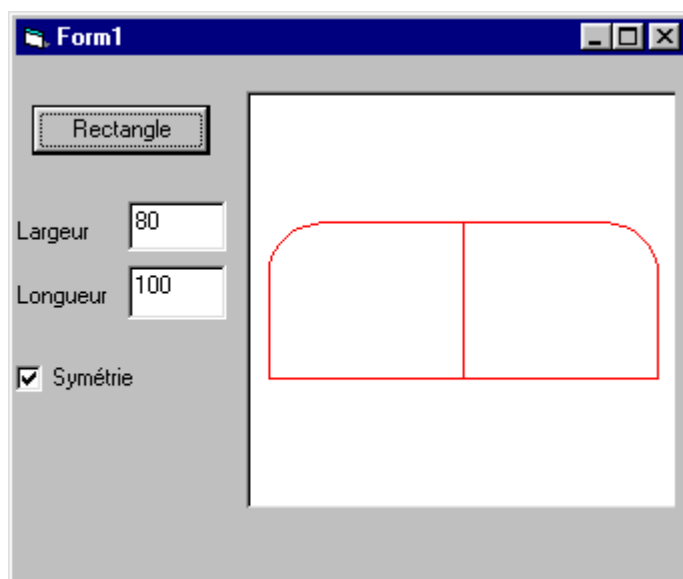
Modifier la propriété *Caption* par le texte *Symétrie*.

Dans la fonction *Command1_Click*, Ajouter et modifier les lignes suivantes.

Call ParamX1.Arrondi(index1, index2, largeur / 3)	
If Check1.Value = 1 Then	Si le bouton symétrie est coché
Dim nbentite As Long	Nombre d'entité dans le dessin
Dim i As Long	Indice de boucle
nbentite = ParamX1.compte	Récupérer le nombre d'entité
For i = 3 To nbentite + 2	Boucle sur toutes les entités
Call ParamX1.SymParAxe(longueur, 0, 90, i, True)	Symétrie / axe vertical x=longueur
Next i	Retour Boucle
End If	Fin du cas symétrie




Résultat :



Formation 4

Amélioration de l'interface

Bouton Quitter

- Ajouter un contrôle *CommandButton*  sur la forme *Form1*.
- Changer sa propriété *Caption* par le texte *Quitter*.
- Double Cliquer sur le bouton.
- Ajouter la ligne suivante à la fonction *Command2_Click*

Private Sub Command2_Click()	
Unload Me	Désallouer toutes les variables et arrêter l'exécution du programme.
End Sub	

Modification Immédiate (Evénements)

On se propose d'obtenir immédiatement le résultat de la modification d'un paramètre.

Pour cela il suffit de faire exécuter la fonction *Command_Click* pour chaque événement pour lequel on souhaite voir le résultat immédiat.

Au démarrage du programme:

Double cliquer sur un endroit libre de la forme *Form1* :

Puis ajouter la ligne suivante :

Private Sub Form_Load()	
Call Command1_Click	Appeler la fonction Command1_Click pour créer la forme avec les valeurs par défaut
End Sub	

Après avoir modifier la case à cocher :

Double cliquer sur le contrôle *Check1* (*Symétrie*)

L'événement par défaut proposé est le *Click* :

Private Sub Check1_Click()	
Call Command1_Click	Appeler la fonction Command1_Click pour modifier le dessin avec la valeur saisie
End Sub	

Après la saisie d'une dimension :

Double cliquer sur un contrôle *TextBox* (*Text1* pour *largeur* par ex.:

L'événement par défaut proposé est le *Change*.

Changer et choisir l'événement *LostFocus* :

Private Sub Text1_LostFocus()	
Call Command1_Click	Appeler la fonction Command1_Click pour modifier le dessin avec la valeur saisie
End Sub	

Idem pour *Text2* .

Contrôle de la saisie

On se propose de contrôler la saisie des dimensions.

Par exemple si la largeur est supérieure à la longueur, afficher un message d'avertissement et empêcher cette valeur.

Pour cela il suffit de modifier, l'événement *LostFocus* du contrôle *Text1* (*largeur*)

Private Sub Text1_LostFocus()	
If (Val(Text1.Text) > Val(Text2.Text)) Then	Si largeur est supérieur à longueur
MsgBox ("Largeur > Longueur")	Afficher le message
Text1.Text = Str(largeur)	Changer la valeur par la précédente
Text1.SetFocus	Mettre le curseur de saisie sur <i>Text1</i> (<i>largeur</i>)
Else	Sinon
Call Command1_Click	Appeler la fonction Command1_Click pour modifier le dessin avec la valeur saisie
End If	
End Sub	

Les Assistants

Introduction

Les assistants XDK permettent de préparer un projet de paramétrage ou de visualisation avec Visual Basic.

Ils créent :

- la forme principale avec
 - les contrôles XDK correspondants
 - quelques boîtes de saisie (Textbox)
 - une barre de menu (fichier, affichage,...)
 - des cadres de résultats (frame)

- le code pour l'initialisation et la saisie des variables

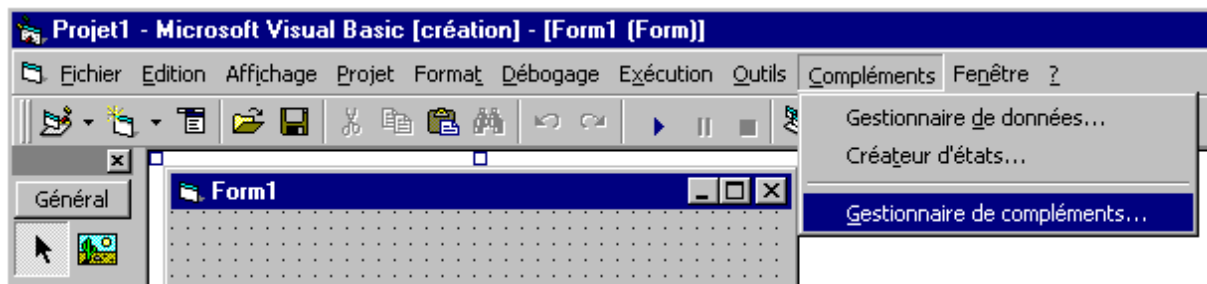
- le code pour la reconstitution du dessin modèle.

Il ne reste plus à l'utilisateur qu'à personnaliser ses boîtes de dialogues et modifier le code du dessin modèle en fonction des paramètres et variables.

Installation des Assistants

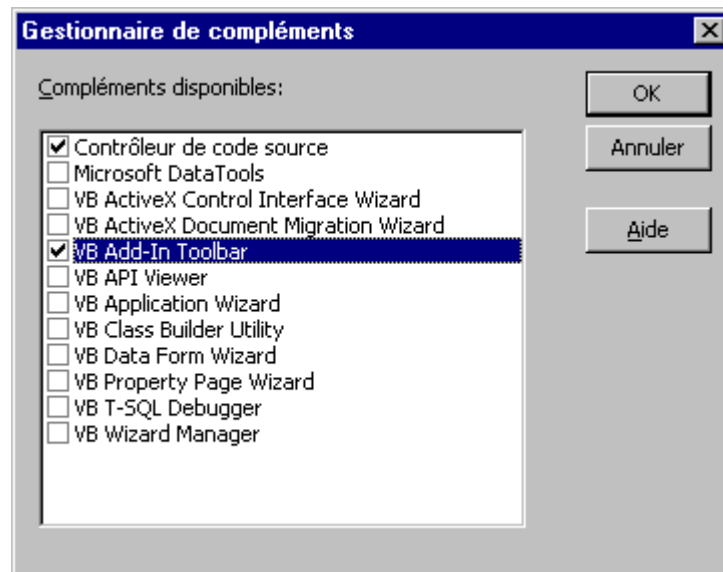
Dans Visual Basic :

Sélectionner les menus suivants

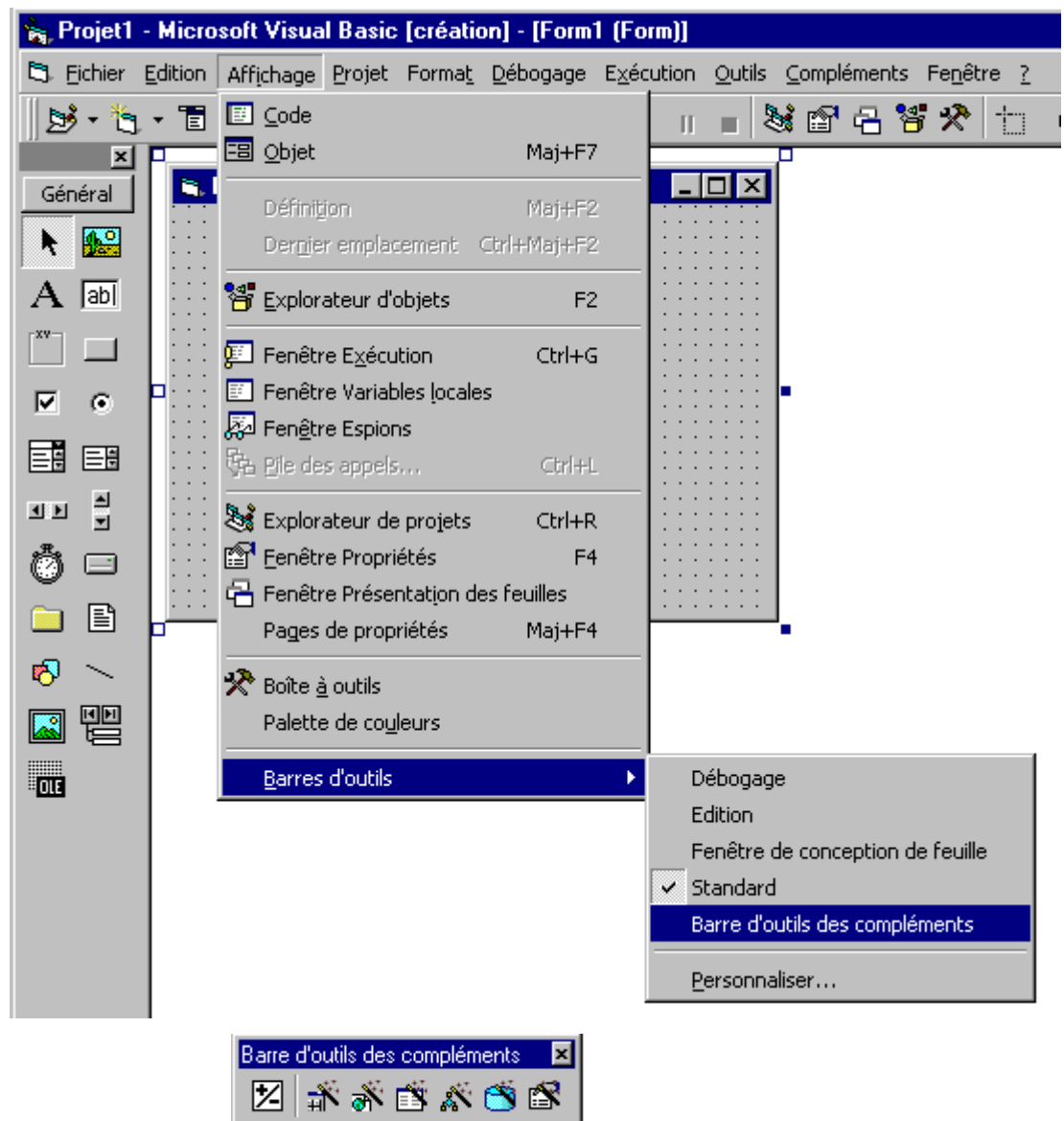


Pour obtenir la barre d'outils des compléments

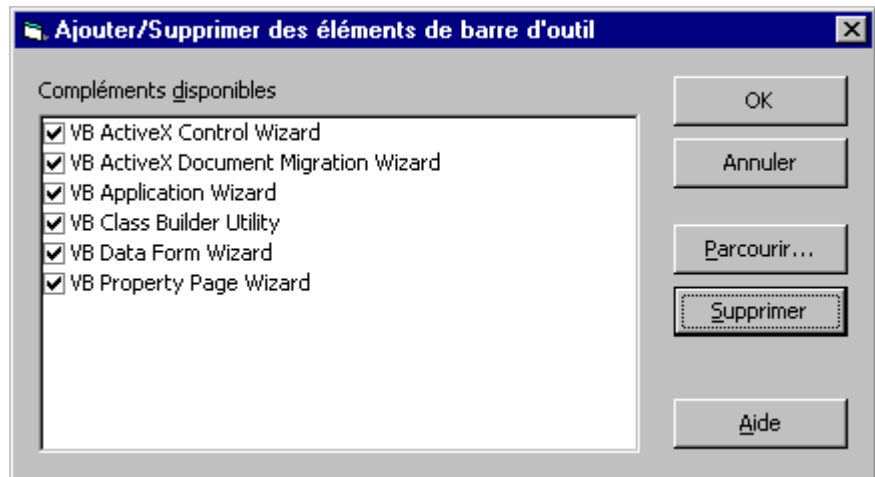
Sélectionner VB Add-In Toolbar



Pour afficher la barre d'outils des compléments.



Utiliser l'icône  pour ajouter un assistant :

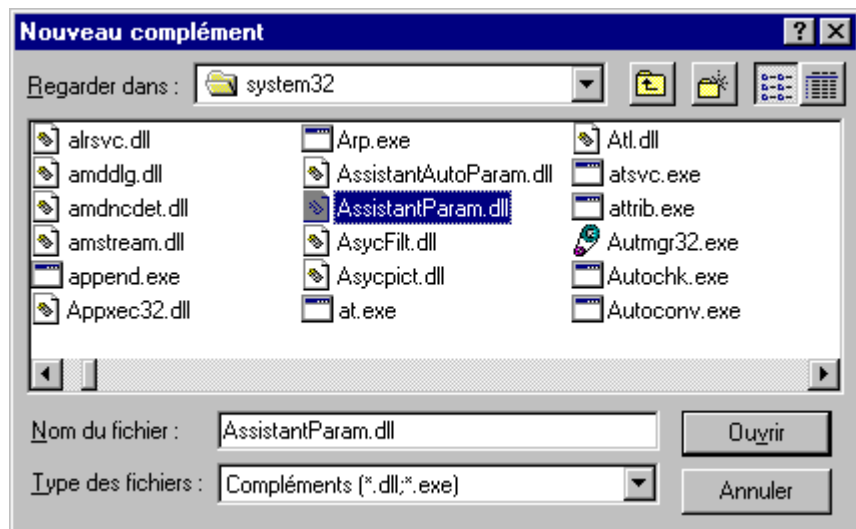


Valider Parcourir :

Pour choisir un assistant dans le répertoire :

c:\windows\system (win95, 98)


c:\winnt\system32 (win NT)

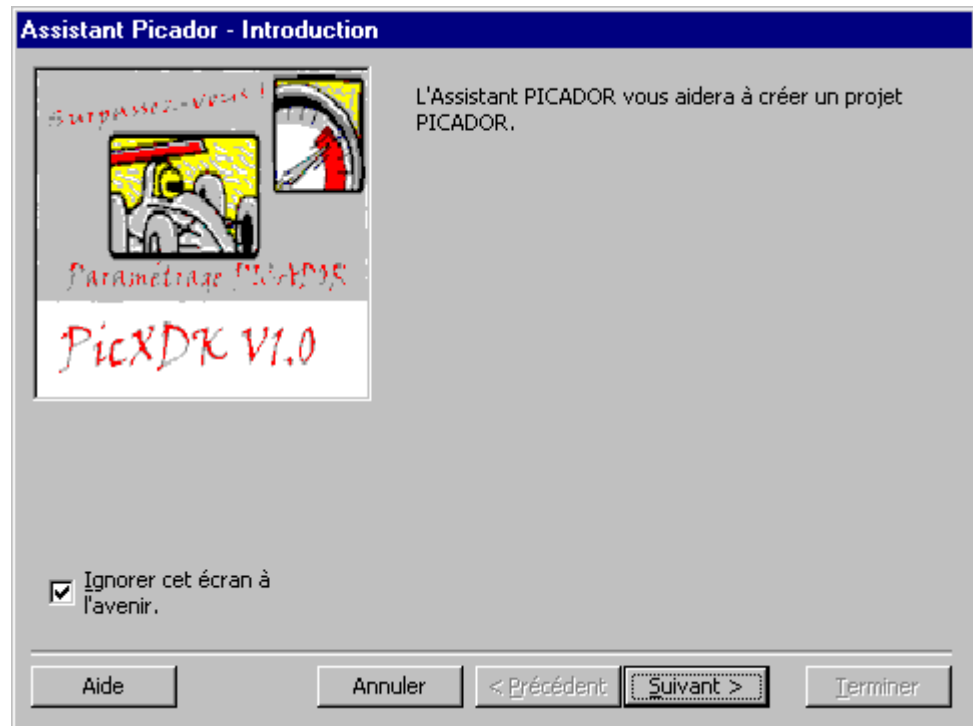


Un nouvel assistant se rajoute à la barre d'outils des compléments :

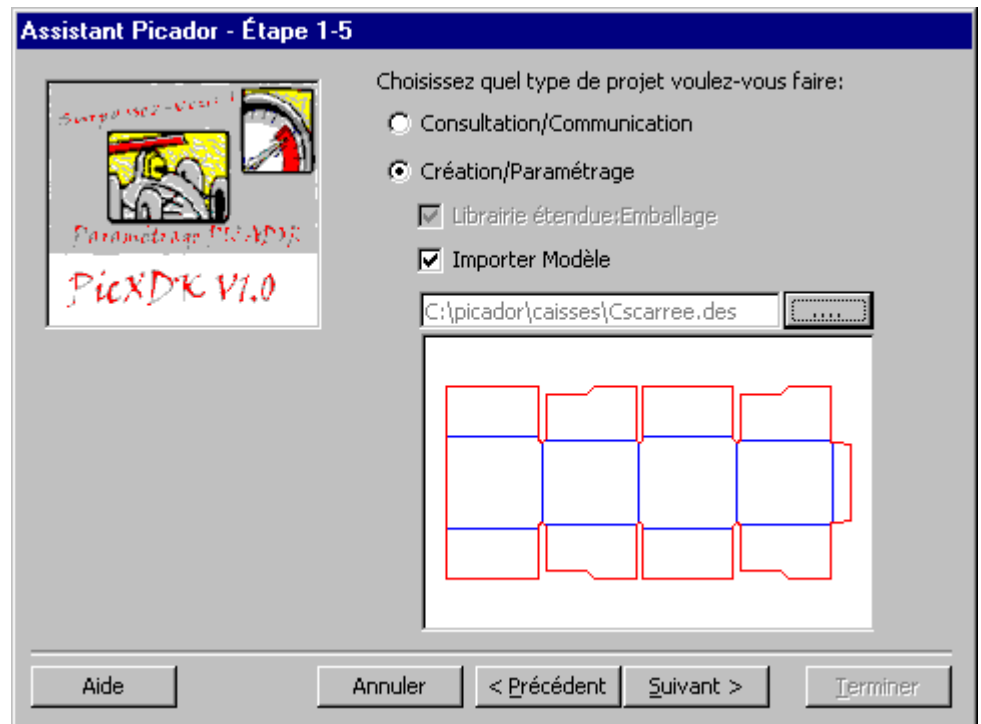


Utilisation des Assistants

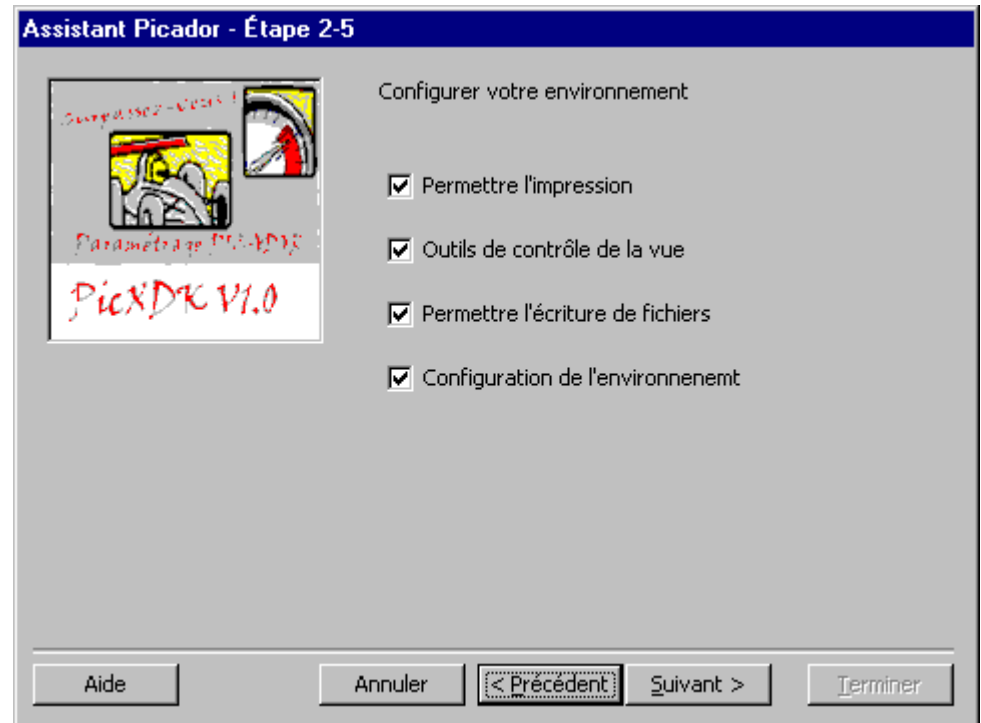
Valider l'icône  dans la barre d'outils des compléments



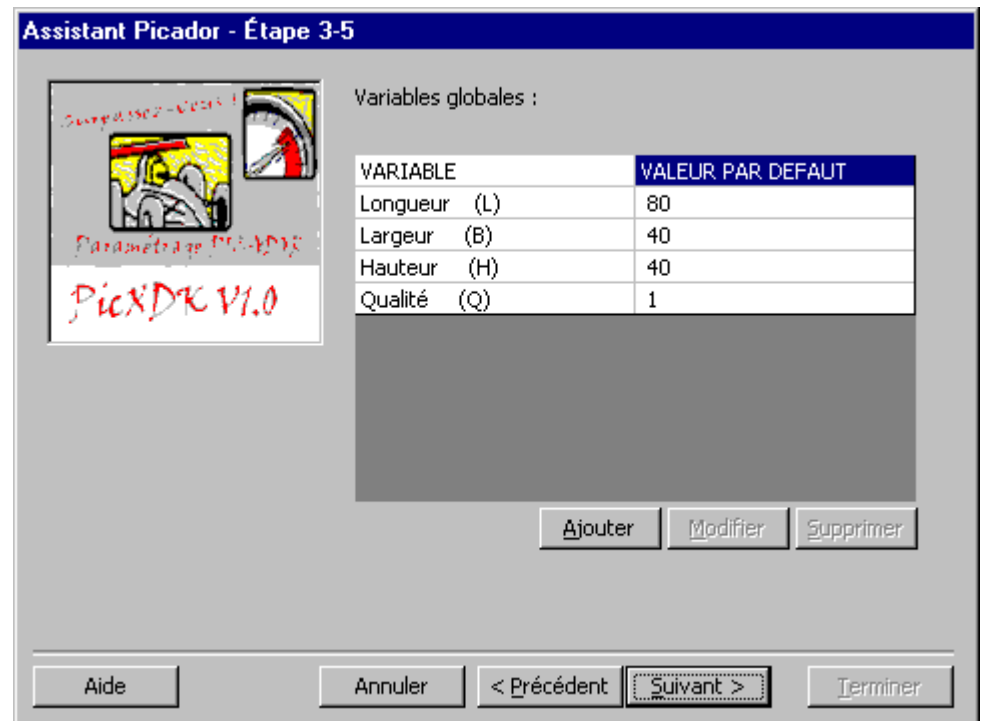
Choisir le dessin modèle permettant de réaliser le projet



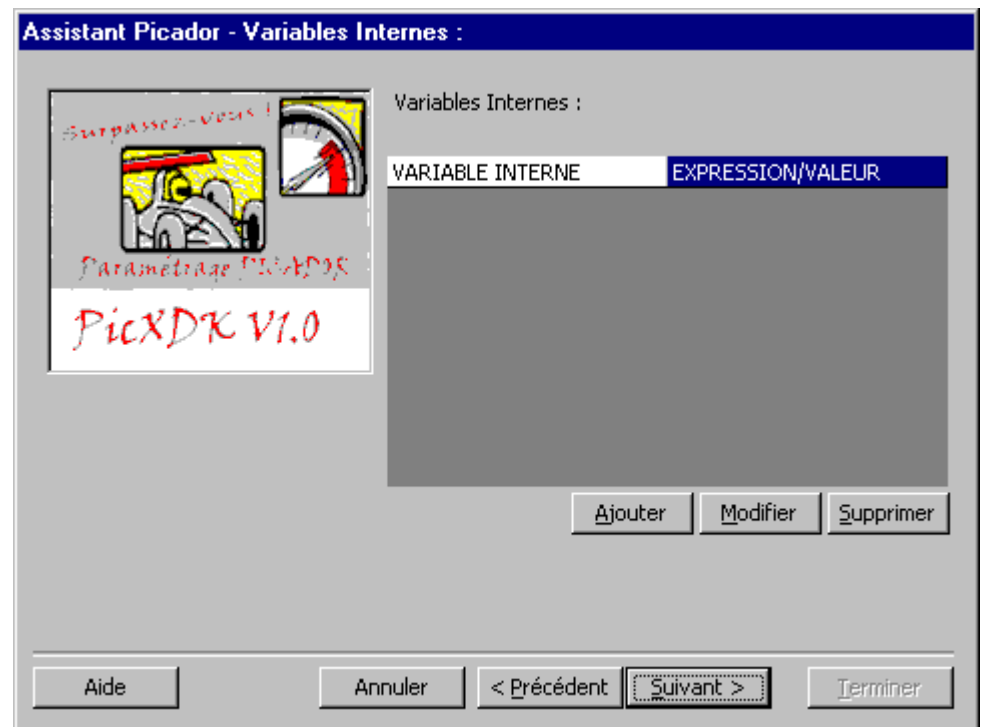
Sélectionner les fonctionnalités dont vous souhaitez disposer dans la boîte de dialogue principale.



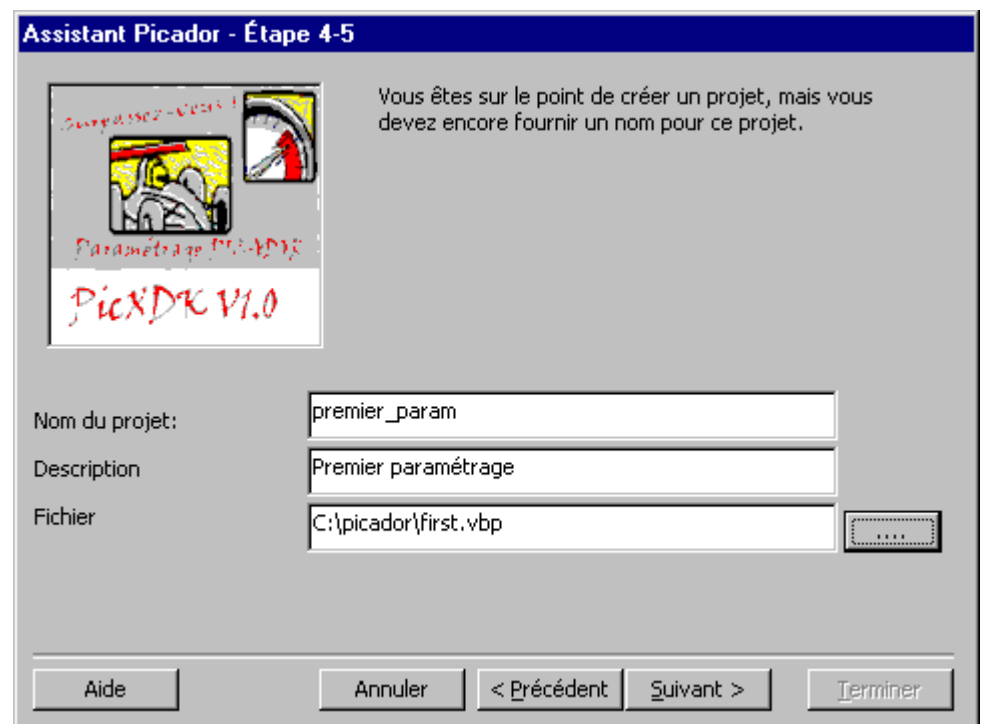
Définition des variables Paramètres et valeur par défaut.



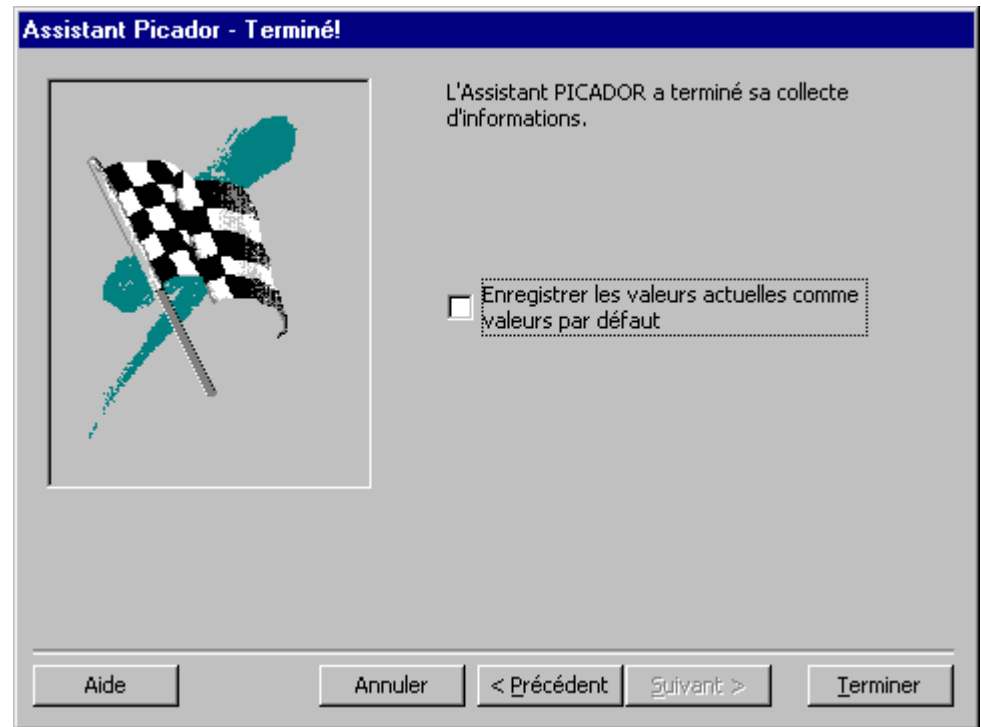
Ajouts de variables internes :



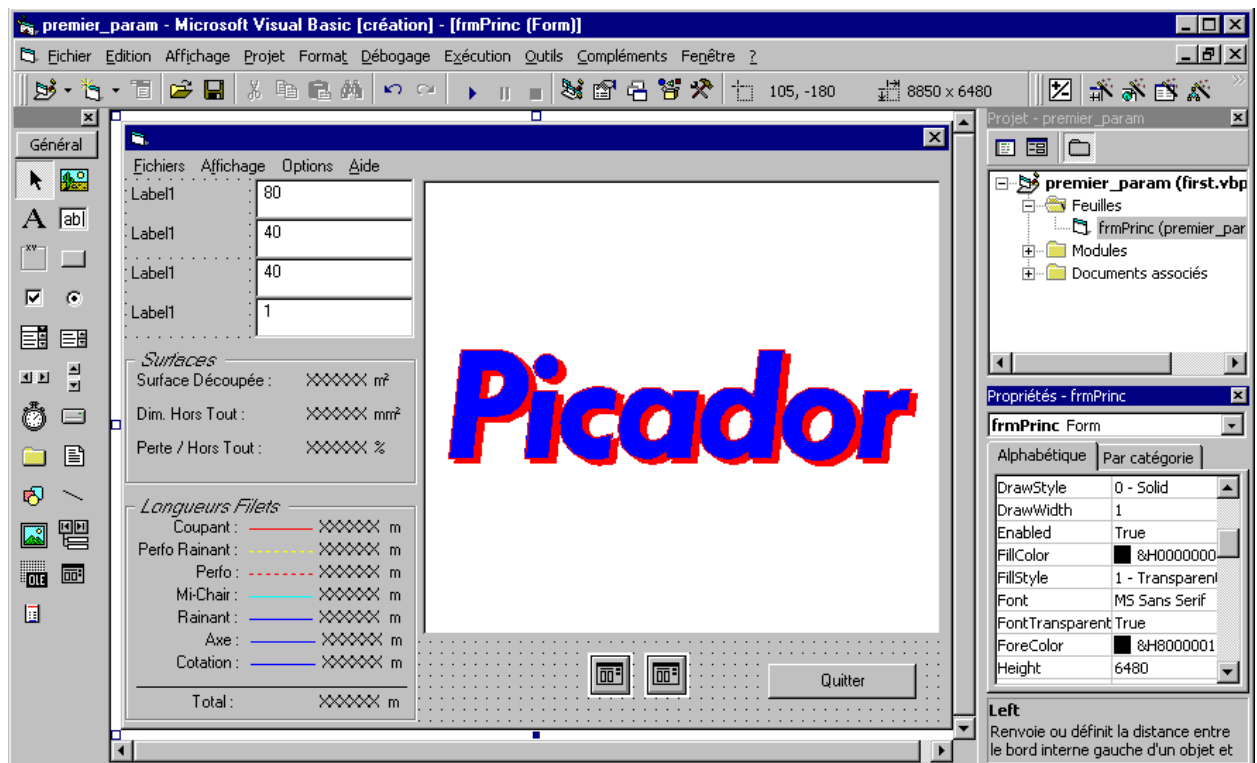
Nom du projet VB et nom du fichier pour l'enregistrer.



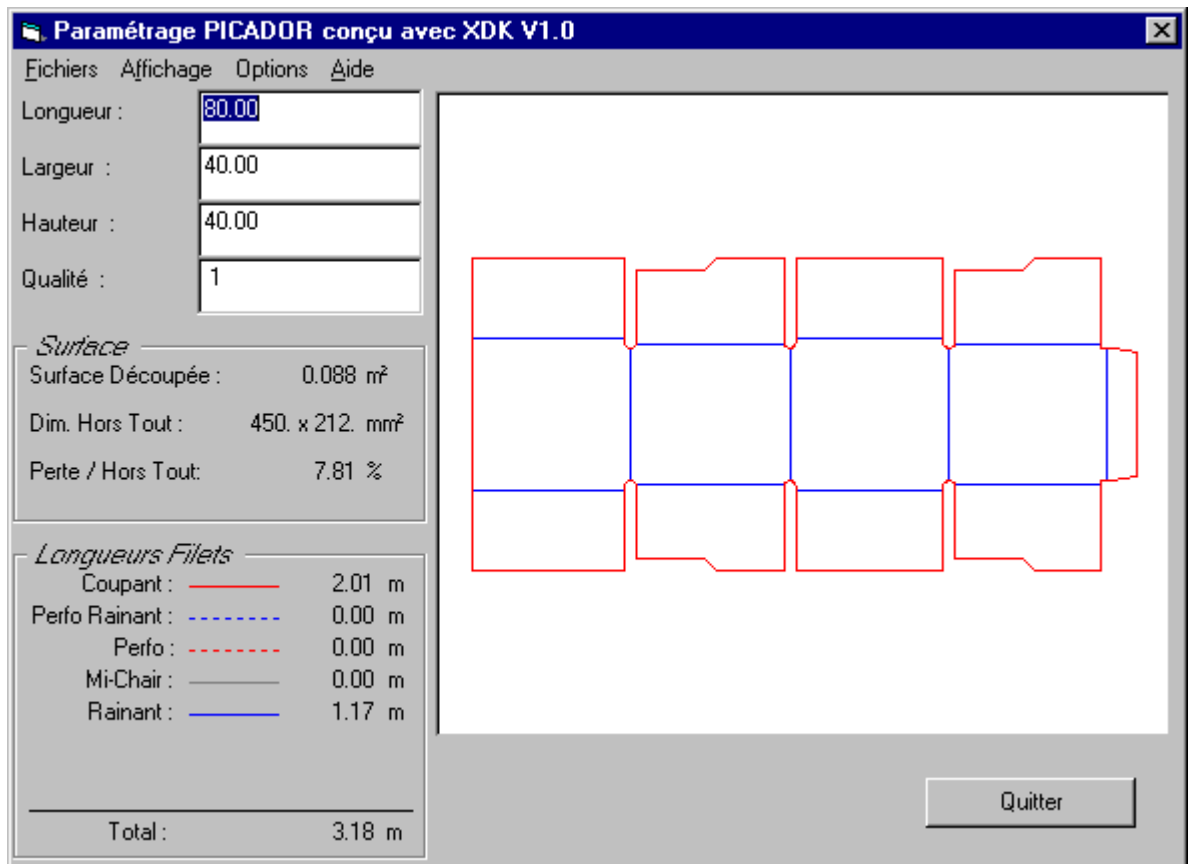
Le projet est créé !



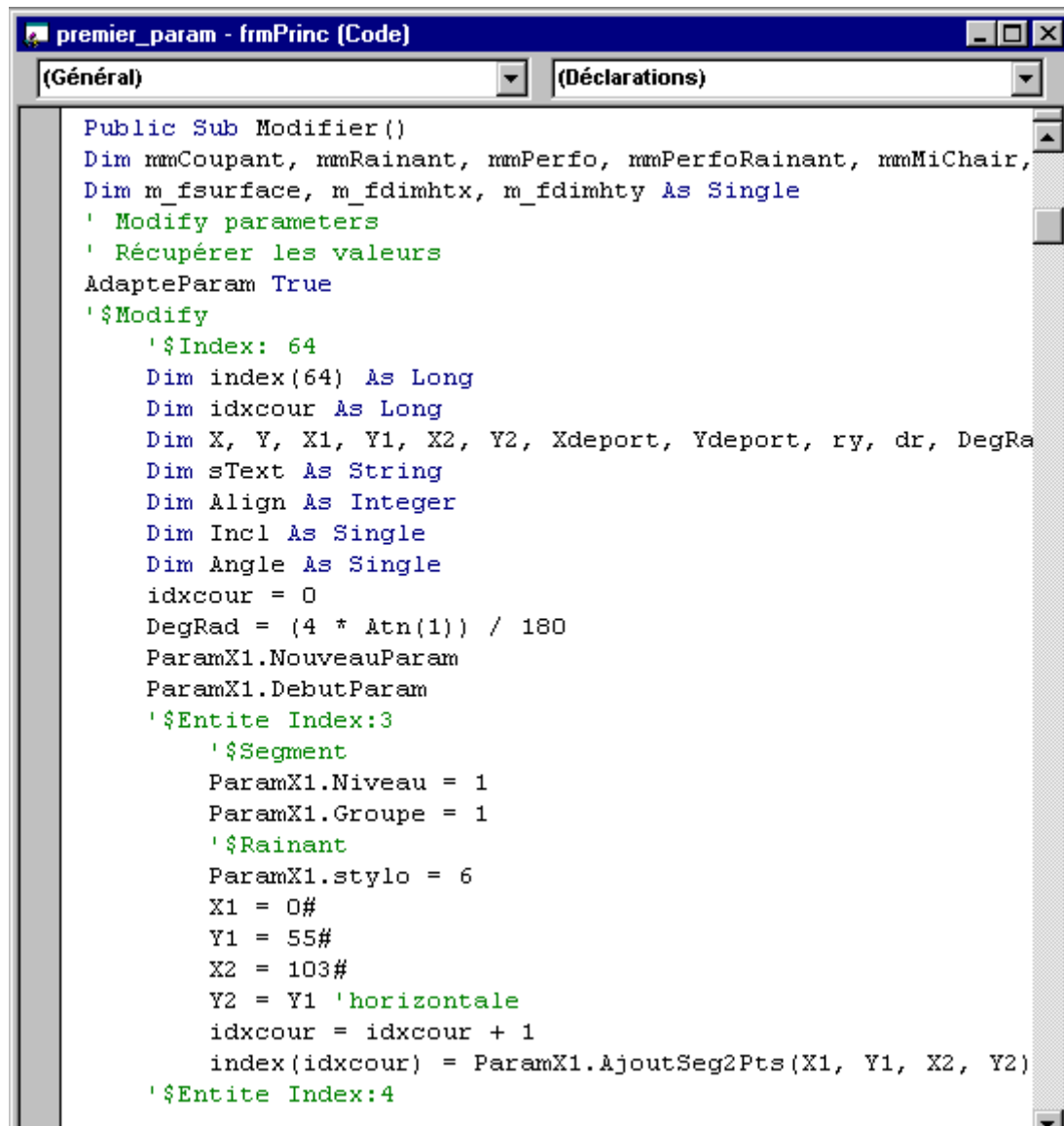
Le projet se présente alors sous la forme suivante :



Il peut être exécuter immédiatement.
Le dessin modèle apparaît avec les dimensions d'origine.



Il suffit ensuite de modifier le code généré automatiquement à l'aide des variables et de personnaliser ses boîtes de dialogues.

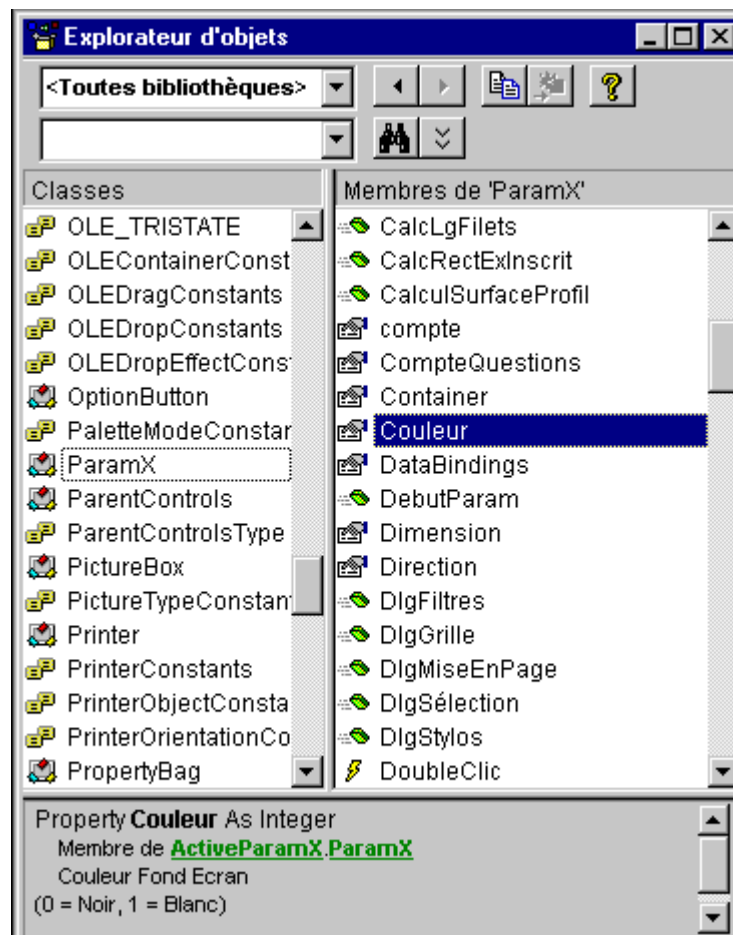


```
Public Sub Modifier()  
Dim mmCoupant, mmRainant, mmPerfo, mmPerfoRainant, mmMiChair,  
Dim m_fsurface, m_fdimhtx, m_fdimhty As Single  
' Modify parameters  
' Récupérer les valeurs  
AdapteParam True  
'$Modify  
    '$Index: 64  
    Dim index(64) As Long  
    Dim idxcour As Long  
    Dim X, Y, X1, Y1, X2, Y2, Xdeport, Ydeport, ry, dr, DegRa  
    Dim sText As String  
    Dim Align As Integer  
    Dim Incl As Single  
    Dim Angle As Single  
    idxcour = 0  
    DegRad = (4 * Atn(1)) / 180  
    ParamX1.NouveauParam  
    ParamX1.DebutParam  
    '$Entite Index:3  
        '$Segment  
        ParamX1.Niveau = 1  
        ParamX1.Groupe = 1  
        '$Rainant  
        ParamX1.stylo = 6  
        X1 = 0#  
        Y1 = 55#  
        X2 = 103#  
        Y2 = Y1 'horizontale  
        idxcour = idxcour + 1  
        index(idxcour) = ParamX1.AjoutSeg2Pts(X1, Y1, X2, Y2)  
    '$Entite Index:4
```

Liste des Propriétés

L' Explorateur d'objets

L'explorateur d'objet permet de consulter l'ensemble des propriétés associé au contrôle XDK.



Compte

Public Property **compte** As Long
Lecture seule
Compteur d'entité (pel dans la version V8)
compteur_entite = Paramx1.compte

CompteQuestions

Public Property **CompteQuestions** As Long
Lecture seule
Compteur de questions
compteur_questions = Paramx1.CompteQuestions

Couleur

Public Property **Couleur** As Integer
Couleur Fond Ecran (0 = Noir, 1 = Blanc)
Paramx1.Couleur = 1

Dimension

Public Property **Dimension** As Single
Dimension courante des entités à créer.
Paramx1.Dimension = 100.0

Direction

Public Property **Direction** As Single
Direction courante des entités à créer (en degré).
Paramx1.Direction = 90.0

DrawLabelNum

Public Property **DrawLabelNum** As Boolean
Bascule Affichage numérotation entité (Vrai/Faux)
Paramx1.DrawLabelNum = TRUE

fichier

Public Property **fichier** As String
Nom du fichier Picador (.des), Autocad (.dxf),
Hpgl (.plt,.hpg)

Paramx1.fichier = "c:\picador\dessin.des "

Grille

Public Property **Grille** As Boolean
Bascule Affichage de la grille (Vrai/Faux)

Paramx1.Grille = FALSE

Groupe

Property **Groupe** As Integer
Numéro de groupe courant des entités à créer

Paramx1.Groupe = 1

LabelQuestion

Property **LabelQuestion**(index As Long) As String
Lecture seule
Question correspondante.

StrQuest = Paramx1.LabelQuestion(idx)

Niveau

Property **Niveau** As Integer
Numéro de niveau courant des entités à créer

Paramx1.Niveau = 1

Origine

Property **Origine** As Boolean
Bascule l'affichage de l'origine (TRUE/FALSE)

Paramx1.Origine = FALSE

Outilvisu

Property **Outilvisu** As String
Permet de définir l'outil de visualisation

Paramx1.Outilvisu = « c:\wpicador\bin32\picview.exe »

Recadrer

Property **Recadrer** As Boolean
Re

Paramx1.Recadrer = TRUE

Librairie des Fonctions Membres

AjoutArc

Ajouter un arc de cercle à la base de donnée.

Déclaration :

```
Public Function AjoutArc (ByVal X As Single, ByVal Y As Single, ByVal ang_o As Single) As Long
```

X, Y = centre de l'arc de cercle

ang_o = angle d'ouverture

AjoutCoteAngle

Ajouter une cote d'angle entre 2 droites à la base de donnée.

Déclaration :

```
Function AjoutCoteAngle(IdxSeg1 As Long, IdxSeg2 As Long, xpos As Single, ypos As Single) As Long
```

IdxSeg1 = Index de la première droite

IdxSeg2 = Index de la deuxième droite

xpos,ypos = position de la cote

AjoutCoteDiametre

Ajouter une cote diamètre intérieure ou extérieure à la base de donnée.

Déclaration :

```
Function AjoutCoteDiametre(IndexArc As Long, xdeport As Single, ydeport As Single, bExterieur As Boolean) As Long
```

IndexArc = Index de l'arc de cercle

xdeport, ydeport = déport extérieur de la cote

bExterieur = TRUE (extérieur) FALSE (intérieur)

AjoutCoteDistance2Pts

Ajouter une cote distance entre 2 points à la base de donnée.

Déclaration :

Function **AjoutCoteDistance2Pts**(x1 As Single, y1 As Single, x2 As Single, y2 As Single, xdeport As Single, ydeport As Single) As Long

x1,y1 = premier point
x2,y2 = deuxieme point
xdeport, ydeport = déport de la cote

AjoutCoteRayonArc

Ajouter une cote rayon à la base de donnée.

Déclaration :

Function **AjoutCoteRayonArc**(IndexArc As Long, xdeport As Single, ydeport As Single, bExterieur As Boolean) As Long

IndexArc = Index de l'arc de cercle
xdeport, ydeport = déport extérieur de la cote
bExterieur = TRUE (extérieur) FALSE (intérieur)

AjoutFormatCarton

Ajouter un format carton à la base de donnée.

Déclaration :

Function **AjoutFormatCarton**(X As Single, Y As Single, x1 As Single, y1 As Single, t As Integer) As Long

X, Y = dimension du carton / marge gauche et bas
x1,y1 = marge gauche et bas / marge droite et haut
t = 0 (dimension connue) / 1 (marges)

AjoutFormatCartonCentre

Ajouter à la base de donnée un format carton centré.

Déclaration :

Function **AjoutFormatCartonCentre**(X As Single, Y As Single, t As Integer) As Long

X, Y = dimension du carton / marges gauche, droite et haut bas
t = 0 (dimension connue) / 1 (marges)

AjoutPose

Ajouter une pose à la base de donnée.

Déclaration :

Function **AjoutPose**(X As Single, Y As Single, dir As Single, grp As Integer, dx As Single, dy As Single, mir As Integer) As Long

X, Y = un point de la pose
dir = rotation de la pose
grp = numéro groupe (Modèle)
dx,dy = translation de la pose par rapport au modèle
mir = miroir de la pose

AjoutProfilFenetre

Ajouter un profil par fenêtre à la base de donnée.

Déclaration :

Function **AjoutProfilFenetre**(xtop As Single, ytop As Single, xbottom As Single, ybottom As Single) As Long

xtop,ytop = angle supérieur de la fenêtre
xbottom,ybottom = angle inférieur de la fenêtre

AjoutProfilPt

Ajouter un profil par point intérieur à la base de donnée.

Déclaration :

Function **AjoutProfilPt**(X As Single, Y As Single) As Long

X,Y = point intérieur

AjoutQuestion

Ajouter un texte-question à la base de donnée.

Déclaration :

Function **AjoutQuestion**(X As Single, Y As Single, question As String, align As Integer, incl As Single) As Long

X,Y = point d'accrochage du texte
question= texte de la question
align = justification (gauche, centre, droite)
incl = inclinaison (italique)

AjoutSeg

Ajouter un segment à la base de donnée.

Déclaration :

Function **AjoutSeg**(X As Single, Y As Single) As Long

X,Y = milieu du segment

AjoutSeg2Pts

Ajouter un segment à la base de donnée.

Déclaration :

Function **AjoutSeg2Pts**(x1 As Single, y1 As Single, x2 As Single, y2 As Single) As Long

x1,y1 = premier point
x2,y2 = deuxième point

AjoutTexte

Ajouter un texte à la base de donnée.

Déclaration :

Function **AjoutTexte**(X As Single, Y As Single, texte As String, align As Integer, incl As Single) As Long

X,Y = point d'accrochage du texte
texte = texte
align = justification (gauche, centre, droite)
incl = inclinaison (italique)

AjoutSeg

Ajouter un segment à la base de donnée.

Déclaration :

Function **AjoutSeg**(X As Single, Y As Single) As Long

X,Y = milieu du segment

ArcCos

Retourne l'angle en degré par rapport à la valeur de son cosinus.

Déclaration :

Function **ArcCos**(X As Double) As Double

X = valeur du cosinus

ArcSin

Retourne l'angle en degré par rapport à la valeur de son sinus.

Déclaration :

Function **ArcSin**(X As Double) As Double

X = valeur du sinus

Arrondi

Ajouter un arrondi entre deux entités.

Déclaration :

Function **Arrondi**(index1 As Long, index2 As Long, rayon As Single) As Long

Index1 = première entité

Index2 = deuxième entité

rayon = rayon de l'entité

(! pour les arrondis droite-arc et arc-arc, positionner le point définissant le secteur :

SetX(-1,X) ; SetY(-1,Y))

CalcLgFilets

Retourne une longueur de filets.

Déclaration :

Function **CalcLgFilets**(niv As Integer, grp As Integer, stylo As Integer) As Single

niv = filtre niveau (-1 tous les niveaux)

grp = filtre groupe (-1 tous les groupes)

stylo = filtre stylo (-1 tous les stylos)

CalcRectExInscrit

Retourne le format hors tout

Déclaration :

Function **CalcRectExInscrit**(xmin As Single, ymin As Single, xmax As Single, ymax As Single) As Boolean

xmin,ymin = angle inférieur gauche
xmax,ymax = angle supérieur droit

CalculSurfaceProfil

Retourne la surface d'un profil.

Déclaration :

Function **CalculSurfaceProfil**(index As Long) As Single

index = entité profil

DebutParam

Début du paramétrage (accès à la base de données)

Déclaration :

Sub **DebutParam**()

DlgFiltres

Boîte de dialogue des filtres.

Déclaration :

Sub **DlgFiltres**()

DlgMiseEnPage

Boîte de dialogue de la mise en page

Déclaration :

Sub **DlgMiseEnPage**()

DlgStylos

Boite de dialogue des stylos.

Déclaration :

```
Sub DlgStylos()
```

Effacer

Définie l'attribut *effacer* de l'entité à *TRUE* . L'entité n'est plus affichée et ne sera pas enregistrée. On récupère l'entité avec la fonction *Recuperer*.

Déclaration :

```
Function Effacer(index As Long) As Boolean
```

Filtrer

Definie le filtre d'affichage.

Déclaration :

```
Sub Filtrer(filtre As String)
```

filtre = commande de filtre

FinParam

Fin du mode paramétrage (accès à la base de données)

Déclaration :

```
Sub FinParam()
```

GetAng_o

Retourne la valeur de l'angle d'ouverture (arc de cercle, ellipse, cote angle).

Déclaration :

Function **GetAng_o**(index As Long) As Single

index = index de l'entité

GetCode

Retourne le code de l'entité

Déclaration :

Function **GetCode**(index As Long) As Integer

index = index de l'entité

GetDeport

Retourne la valeur de départ de l'entité (cotation)

Déclaration :

Function **GetDeport**(index As Long) As Single

index = index de l'entité

GetDim

Retourne la dimension de l'entité

Déclaration :

Function **GetDim**(index As Long) As Single

index = index de l'entité

GetDir

Retourne la direction de l'entité

Déclaration :

Function **GetDir**(index As Long) As Single

index = index de l'entité

GetGroupe

Retourne le groupe de l'entité

Déclaration :

```
Function GetGroupe(index As Long) As Integer
```

index = index de l'entité

GetNiveau

Retourne le code de l'entité

Déclaration :

```
Function GetNiveau(index As Long) As Integer
```

index = index de l'entité

GetRGB

Retourne la couleur correspondant au stylo

Déclaration :

```
Function GetRGB(i As Integer, blmpression As Boolean) As Long
```

i = stylo

blmpression = (TRUE) à l'imprimante (FALSE) à l'affichage

GetStylo

Retourne le stylo de l'entité

Déclaration :

```
Function GetStylo(index As Long) As Integer
```

index = index de l'entité

GetTexte

Retourne le texte de l'entité

Déclaration :

Function **GetTexte**(index As Long) As String

index = index de l'entité

GetX

Retourne la coordonnée en X de l'entité

Déclaration :

Function **GetX**(index As Long) As Single

index = index de l'entité

GetY

Retourne la coordonnée en Y de l'entité

Déclaration :

Function **GetY**(index As Long) As Single

index = index de l'entité

Imprimer

Imprime le dessin associé au contrôle.

Déclaration :

Sub **Imprimer**(JobName As String)

JobName = nom du job à l'imprimante

ImprimerDirect

Definie le filtre d'affichage.

Déclaration :

Sub **ImprimerDirect**(orientation As Integer, FormatPapier As Integer, nbcopies As Integer, qualite As Integer, RectoVerso As Integer)

InsererModFT

Insère dans le document courant une fiche technique et son questionnaire .

Déclaration :

Sub **InsererModFT**(fichier As String)

fichier = nom du modele de fiche technique.

Les réponses au questionnaire sont créées avec un texte vide.

nomModFT

Retourne le nom du modèle de fiche technique.

Déclaration :

Function **nomModFT**() As String

Ouvrir_Avec

Définit l'application permettant de visualiser ou de modifier le document.

Déclaration :

Sub **Ouvrir_Avec**()

PtsInterEntites

Définit le point d'intersection entre 2 entités.

Déclaration :

Function **PtsInterEntites**(xdecision As Single, ydecision As Single, dr1 As Single, dr2 As Single, index1 As Long, index2 As Long, x1 As Single, y1 As Single, x2 As Single, y2 As Single) As Boolean

xdecision, ydecision = point permettant d'ordonner les points d'intersections

dr1, dr2 = translation des entités 1 et 2

index1, index2 = index des 2 entités

x1,y1 = intersection la plus proche de xdecision, ydecision

x2,y2 = deuxième intersection

Recuperer

Recuperer une entité effacée.

Déclaration :

Function **Recuperer**(index As Long) As Boolean

index = index de l'entité

Refresh

Rafraîchir le contrôle.

Déclaration :

Sub **Refresh**()

ReIndex

Reindexe la base de données

Déclaration :

Sub **ReIndex**()

SauverSous

Sauvegarder le dessin dans un fichier.

Déclaration :

Sub **SauverSous**(fichier As String)

fichier = nom du fichier (.des, .dxf, .hpg, .ai, .n, .cf2)

SetAng_o

Definie l'angle d'ouverture de l'entité.

Déclaration :

Function **SetAng_o**(index As Long, ang As Single) As Integer

index = index de l'entité
ang = angle d'ouverture

SetDeport

Définie le déport de la cotation.

Déclaration :

Function **SetDeport**(index As Long, dep As Single) As Integer

index = index de l'entité
dep = déport

SetDim

Définie la dimension de l'entité

Déclaration :

Function **SetDim**(index As Long, d As Single) As Integer

index = index de l'entité
d = dimension

SetDir

Définie la direction de l'entité.

Déclaration :

Function **SetDir**(index As Long, dr As Single) As Integer

index = index de l'entité
dr = direction

SetGroupe

Définie le groupe de l'entité.

Déclaration :

Function **SetGroupe**(index As Long, grp As Integer) As Integer

index = index de l'entité
grp = groupe

SetNiveau

Définie le niveau de l'entité.

Déclaration :

Function **SetNiveau**(index As Long, niv As Integer) As Integer

index = index de l'entité
niv = niveau

SetStylo

Définie le stylo de l'entité.

Déclaration :

Function **SetStylo**(index As Long, sty As Integer) As Integer

index = index de l'entité
sty = stylo

SetTexte

Définie le texte de l'entité.

Déclaration :

Function **SetTexte**(index As Long, txt As String) As Integer

index = index de l'entité
txt = texte

SetX

Définie la coordonnée en X de l'entité.

Déclaration :

Function **SetX**(index As Long, X As Single) As Integer

index = index de l'entité
X = coordonnée en X

SetY

Définit la coordonnée en Y de l'entité.

Déclaration :

Function **SetY**(index As Long, Y As Single) As Integer

index = index de l'entité
Y = coordonnée en Y

SymParAxe

Symétrie de l'entité par rapport à un axe .

Déclaration :

Function **SymParAxe**(X As Single, Y As Single, dir As Single, index1 As Long, bCopie As Boolean) As Long

X,Y = un point de l'axe de symétrie
dir = direction de l'axe de symétrie
Index1 = index de l'entité
bCopie = (TRUE) ajoute l'entité symétrique (FALSE) transforme l'entité

Tranf2d

Transformation 2D (translation, rotation, homothétie) d'un bloc d'entité

Déclaration :

Sub **Tranf2d**(index1 As Long, index2 As Long, X As Single, Y As Single, dir As Single, hom As Single)

Index1 = index début du bloc d'entités
Index2 = index fin du bloc d'entités
X,Y = vecteur de translation
dir = rotation
hom = rapport d'homothétie (si <0 : symétrie)

ZoomArrière

Fonction Zoom arrière

Déclaration :

Sub **ZoomArrière**()

ZoomAvant

Fonction Zoom Avant

Déclaration :

Sub **ZoomAvant**()

ZoomPan

Fonction Zoom panoramique.

Déclaration :

Sub **ZoomPan**()
