MACRO-CODE UTILISATEUR (MCU)

PRESENTATION

Les Macro-Codes Utilisateur sont des morceaux de code Windev que vous tapez et qui permettent de définir vos propres traitements dans votre application.

Vous pouvez ainsi par exemple :

- Ajouter une demande de confirmation lors d'un clic sur un bouton, pour éviter qu'un traitement sensible ne soit lancé par erreur.
- Dans une table, changer la couleur de certaines lignes. Par exemple, dans la fenêtre de gestion de salariés, les salariés sont présentés en rouge ou en bleu en fonction de l'établissement auquel ils sont affectés.

La saisie de ce code est simplifiée grâce à un assistant, mais elle demande un niveau de connaissances minimal du langage Windev utilisé pour « rédiger » la macro. Une présentation de ce langage vous est proposée en annexe de ce document. Prenez quelques minutes pour le découvrir. Au besoin, faites vous aider par un informaticien connaissant Windev !

Lorsqu'une fenêtre permet la saisie de Macro-Code Utilisateur, une icône supplémentaire apparaît dans la barre de titre (sous Windows XP seulement), à coté de l'icône de réduction (ou de fermeture) de la fenêtre (en haut, à droite de la fenêtre). Lors du clic sur cet icône, un assistant apparaît, et vous permet de créer, modifier ou supprimer vos Macro-Code Utilisateur. Si cet icône n'apparait pas, il est toujours possible d'accéder à la fenêtre de gestion des MCU par le raccourci clavier *Windows + F2*.

ACTIVATION

Cette fonctionnalité MCU n'est pas activée par défaut. Sa mise en œuvre se fait au travers de la gestion des sécurités, avec deux niveaux d'accès : utilisation de MCU existant uniquement, ou gestion complète du MCU, autorisant donc la saisie de nouveaux Macro-Codes.

Pour activer le MCU, il faut :

- créer un domaine de sécurité nommé *MCU* ;
- accorder des droits d'accès à ce domaine *MCU*, société par société, aux différents utilisateurs concernés. Si aucun droit d'accès n'est donné, ou si un droit d'accès *Restreint* est accordé, le MCU est interdit (le bouton *MCU* n'apparaît pas dans la barre de titre des fenêtres) ; si l'accès est *Partiel*, on peut exécuter les MCU existants, mais l'accès à la fenêtre de gestion du MCU (par le clic sur le bouton *MCU* de la barre de titre, ou la touche *Windows F2*) est bloqué ; enfin, si l'accès est *Complet* ou *Administrateur*, on peut créer/ajouter/supprimer du MCU dans toutes les fenêtres.

AJOUTER UN MACRO-CODE UTILISATEUR

Pour ajouter un Macro-Code Utilisateur :

1. Lancez l'assistant par un clic sur l'icône *MCU* dans la barre de titre de la fenêtre, ou combinaison de touches *Windows + F2*.
2. Cliquez sur le bouton *Ajouter un Macro-Code*.
3. Indiquez le type d'action concerné par le traitement personnalisé : Action sur un champ ou sur une combinaison de touches.
4. Si vous effectuez un traitement sur un champ :

- Sélectionnez le champ concerné. Un tableau liste les différents champs de l'application avec leur libellé. Vous pouvez également sélectionner directement le champ dans la fenêtre grâce au bouton "Choisir le champ".
- Sélectionnez l'action à personnaliser, et le moment de l'exécution de l'action personnalisée.
- Un éditeur de code simplifié apparaît, permettant de saisir le titre de la macro, et le code W-Langage du Macro-Code Utilisateur.

5. Si vous effectuez un traitement sur une combinaison de touches :
   - Spécifiez la combinaison de touches concernée.
   - Un éditeur de code simplifié apparaît, permettant de saisir le titre de la macro, le moment d'exécution et le code W-Langage du Macro-Code Utilisateur.

6. Dans l'éditeur de code simplifié :
   - la coloration syntaxique est disponible.
   - l'aide des fonctions W-Langage est disponible (si vous l'avez installée, voir annexe 2).
   - Il est possible de tester directement le code saisi.
   - Il est possible d'obtenir la liste des champs, afin de manipuler un autre champ de la fenêtre.

7. Validez. Le Macro-Code est ajouté et est disponible immédiatement.

Annexe – Présentation du langage Windev

L'objet de ce chapitre est de présenter de façon très succincte la syntaxe du langage Windev, et les fonctions les plus usitées.

Notez qu'une aide en ligne est disponible pour le langage Windev ; il vous faut la télécharger depuis le site de PCSoft, à l'adresse

ftp://framework.pcsoft.fr/HLP/12.0/FR/InstallHlpMCU.EXE

Une fois cette aide installée, dans la fenêtre de saisie d'un Macro-Code Utilisateur, on peut frapper le nom d'une fonction par exemple puis appuyez sur *F1* en ayant positionné le curseur sur le nom de la fonction ; on obtient ainsi l'aide détaillée décrivant le fonctionnement exact de la fonction et les paramètres de celle-ci.

SYNTAXE GENERALE DU LANGAGE

Tout code Windev est constitué d'une suite d'instructions. En règle générale, on saisit une instruction par ligne, mais on peut aussi :

- saisir plusieurs instructions sur une même ligne, en les séparant par un point virgule « ; »
- saisir une instruction en plusieurs lignes ; la ligne doit pour cela se terminer par trois points accolés les uns aux autres « … »

On peut toujours commenter le code que l'on saisit, et cela est même vivement recommandé ; tout ce qui suit une séquence de deux caractères « // » est du commentaire, et sera donc ignoré par le système.

Pour référencer une valeur constante, indiquer directement la valeur s'il s'agit d'une valeur numérique ; placez cette valeur entre guillemets « " » s'il s'agit d'une chaine de caractères.

Le langage Windev est insensible à la casse ; vous pouvez donc indifféremment saisir votre code en majuscule ou en minuscule.

Enfin, les espaces, dès lors qu'ils se trouvent en dehors d'une valeur de type chaine de caractères (toujours frappée entre guillemets `"comme ceci"`) ne sont pas significatifs. Vous pouvez en placer à votre guise.

LES VARIABLES

Si vous avez besoin d'une variable pour enregistrer des résultats intermédiaires de calcul, il faut la déclarer. Une variable se définit par son nom et son type. Le type de la variable détermine les valeurs que la variable peut prendre, et les opérations qui sont possibles. La déclaration d'une variable est toujours de la forme :

NomVariable est un(e) TypeVariable

ou

NomVariable1,NomVariable2 sont des TypeVariables

Remarque : les mots-clés *un*, *une* et *des* ne sont pas obligatoires ; ce sont des mots d'agrément. De même, le type de la variable peut être défini au singulier ou au pluriel ; l'orthographe n'est pas vérifiée par le système ; le respect des règles d'orthographe a simplement pour but de faciliter la relecture du code par une autre personne.

Quelques exemples :

```
Indice est un entier
NomSalarié est une chaine
Montant est un réel
I, J, K sont des entiers
```

Les principaux types sont :

- *Entier*, pour manipuler des valeurs entières uniquement
- *Réel*, pour manipuler des valeurs réelles (avec des décimales)
- *Chaine*, pour manipuler des chaines de caractères

On peut aussi déclarer et initialiser une variable en une seule instruction :

NomVariable est un(e) TypeVariable = Valeur

Exemple :

```
Compteur est un entier = 15
MoisPaye est une chaine = "200703"
```

LES TABLEAUX

Windev sait manipuler un grand nombre de tableaux. : simples, dynamiques, à une ou plusieurs dimensions… Contentons nous de la forme la plus simple :

NomTableau est un(e) tableau de Dimension TypeVariable

Exemple :

```
BrutMensuel est un tableau de 12 réels
```

Pour faire référence à un élément du tableau :

```
NomTableau[Indice]
```

Pour connaître le nombre d'élément d'un tableau :

```
Dimension(NomTableau)
```

Exemple (calcul de la somme des éléments d'un tableau):

```
BrutMensuel est un tableau de 12 réels
TotalAnnuel est un réel
i est un entier
Pour i = 1 à Dimension(BrutMensuel)
   TotalAnnuel = TotalAnnuel + BrutMensuel[i]
Fin
```

LES OPERATEURS

LES OPERATEURS LOGIQUES

Les opérateurs logiques sont *ET*, *OU*, *PAS*. Ces opérateurs logiques permettent de construire des conditions.

Exemple :

```
// Tester qu'une valeur est dans un intervalle donné
SI BrutMensuel < 1000 ET BrutMensuel < 1500 ALORS
```

Les règles de priorité des opérateurs sont classiques ; dans le doute, vous pouvez utiliser des parenthèses pour forcer l'ordre d'évaluation des conditions :

```
SI (Brut < 1000 ET Brut < 1500) OU Catégorie="10" ALORS
```

LES OPERATEURS ARITHMETIQUES

En plus des 4 opérateurs classiques :

+	Addition
-	Soustraction
*	Multiplication
/	Division

on trouve d'autres opérateurs forts pratiques :

++	Incrémentation :	`I++` est équivalent à `I=I+1`
--	Décrémentation :	`I--` est équivalent à `I=I-1`
+=	Ajout d'une valeur	`I+=5` est équivalent à `I=I+5`

Les règles de priorité des opérateurs sont là aussi classiques ; utilisez les parenthèses lorsque cela est nécessaire :

```
I = 5 ; J = 6
I = I + J * 3        // I vaut maintenant 23
I = (I + J) * 3      // I vaut maintenant 33
```

LES OPERATEURS DE COMPARAISON

Ce sont les opérateurs classiques :

=	Egal
<>	Différent
<=	Inférieur ou égal
>=	Supérieur ou égal
<	Inférieur strictement
>	Supérieur strictement

Ces opérateurs peuvent être utilisés tant avec des variables ou valeurs numériques qu'avec des chaines de caractères.

On dispose également d'opérateurs spécifiques aux chaines de caractères :

~=	Egalité souple : ne tient pas compte ni des caractères majuscules ou minuscules, ni des voyelles accentuées, ni des espaces situés avant ou après la chaine de caractères à tester
[=	Commence par

Exemples:

```
"Dupont" = "DUPONT"       // Retourne Faux
"Dupont" ~= "DUPONT"      // Retourne Vrai
```

LES INSTRUCTIONS CONDITIONNELLES

L'instruction la plus utile est *SI, SINON, FIN*.

Quelques exemples :

```
SI Indice < 0 ALORS Indice=0    // La forme la plus simple

SI Indice < 0 ALORS             //  Sur  plusieurs  lignes,  avec
Sinon
   Indice = 0
SINON
   Indice=Indice+1
FIN

SI Indice < 0 ALORS             // Plusieurs conditions en cascade
   Libellé = "Erreur"
SINON SI Indice = 0
   Libellé = "Pas encore traité"
SINON
```

```
   Libellé = "c'est OK"
FIN

SI I < 10 ALORS                    // Plusieurs conditions imbriquées
   SI J < 10 ALORS
         J = J+1
   SINON
         J=0 ; I ++
   FIN
SINON
   Libellé = "c'est fini"
FIN
```

LES INSTRUCTIONS DE BOUCLE

On instructions de boucle les utiles sont :

- *POUR FIN* pour un nombre d'itérations déterminé
- *TANTQUE FIN* pour réaliser un traitement tant qu'une condition n'est pas vérifiée.

Quelques exemples :

```
Indice est un entier
POUR Indice = 1 à 100
   // Traitement ici réalisé 100 fois
FIN

Indice est un entier
Indice=1
TANTQUE Indice <= 100
   Indice ++
   // Traitement ici réalisé aussi 100 fois
FIN
```

LES FONCTIONS DE BASE SUR LES VARIABLES NUMERIQUES, CHAINES, ET DATES

Windev offre plusieurs centaines de fonctions intégrées prêtes à l'emploi. Nous allons en décrire ici seulement quelques unes, d'un usage très fréquent.

MANIPULATION DE CHAINES DE CARACTERES

Pour concaténer deux chaînes de caractères, utiliser l'opérateur **+**.

```
NomPrénom = PEPERS.NMAG + " " + PEPERS.PREN
```

Pour extraire une partie d'une chaine de caractères, plusieurs possibilités :

```
Chaine= »ABCDEF »
NouvelleChaine=Chaine[[2 à 3]]          // NouvelleChaine vaut "BC"
NouvelleChaine=Gauche(Chaine, 2)        // NouvelleChaine vaut "AB"
NouvelleChaine=Droite(Chaine, 2)        // NouvelleChaine vaut "EF"
NouvelleChaine=Milieu(Chaine, 2, 3)     // NouvelleChaine vaut "BCD"
```

La fonction *Taille* permet de connaître la taille d'une chaine de caractères.

```
Info( Taille(Chaine))      // Affiche 5
```

La fonction *Extraitchaine* permet d'extraire une partie d'une chaine composée d'une suite de plusieurs éléments séparés par un caractère bien identifiés.

```
MaListe est une chaine = "01;02;003;004"
Chaine=ExtraitChaine(MaListe,3, ";")  // Chaine vaut "003"
```

Une chaine peut être transformée en majuscule ou en minuscule :

```
Chaine=Majuscule("Abcdef")       // Chaine vaut "ABCDEF"»
Chaine=Minuscule("Abcdef")       // Chaine vaut "abcdef"
```

Une chaine peut être recherchée dans une autre chaine avec la fonction *Position* :

```
MaChaine est une chaine = "ABCDEFGH"
Pos est un entier
Pos=Position(MaChaine,"DE",1)   // Pos vaut 5
// le 1er paramètre définit la chaine dans laquelle on recherche
// le 2ème paramètre définit la chaine recherchée
// le 3ème paramètre définit la position à partir de la quelle on recherche
// On peut également spécifier un 4ème paramètre SansCasse pour
//   que la recherche s'effectue sans tenir compte
//   des majuscules et minuscules
```

Une chaine peut être complétée à une longueur donnée :

```
MaChaine est une chaine = "ABCDEFGH"
MaChaine=Complete(MaChaine,10)  // MaChaine a été complété avec 2 espaces à droite
```

Autres fonctions pouvant s'avérer utiles :

```
MaChaine=Répète("0", 10)  // Contient 10 fois le caractères 0
MaChaine=SansEspace(MaChaine)   // Elimine les espaces situés à gauche                    //    et à droite
```

MANIPULATION DES VALEURS OU VARIABLES NUMERIQUES

Utilisez un point décimal pour les valeurs comportant une partie décimale

```
Salaire est un réel
Salaire = 1345.56          // et non pas 1345,56
```

Tous les opérateurs mathématiques décrits plus haut peuvent être utilisés, avec les règles de priorité classiques. On peut utiliser des parenthèses pour forcer un ordre de calcul particulier :

```
Salaire est un réel = 1234.56
I est un entier = 2
Résultat est un réel
Résultat = Salaire * (I+1) / 10       // Résultat = 370.368
```

Il est possible de concaténer une chaîne et un numérique avec l'opérateur +. Le résultat est une chaîne de caractères :

```
Info("Le résultat est "+Résultat)     // Pour afficher le résultat
```

Autres fonctions utiles (qui se passent d'explications) :

```
R est un réel = -1234.56
P est un réel
P=Abs(R)                    // P vaut 1234.56
P=PartieEntière(P)          // P vaut 1234
P=PartieDécimale(R)         // P vaut -0.56
P=Arrondi(R,1)              // P vaut -1234,6 // 0.56 arrondi à 0.60
```

Pour transformer une valeur numérique en chaîne de caractères, utilisez la fonction *NumériqueVersChaine* :

```
R est un réel = 1234.56
Info(NumériqueVersChaine(R, "10,3f")) // Affiche 1234,560
Info(NumériqueVersChaine(R, "010.3f")) // Affiche 001234.560
I est un entier = 1234
Info(NumériqueVersChaine(I)) // Affiche 1234
Info(NumériqueVersChaine(I, "05d")) // Affiche 01234
```

A l'inverse, pour transformer une chaîne en valeur numérique, utilisez la fonction *Val* :

```
MaChaine est une chaine = "1234.56"
R est un réel
R=Val(MaChaine) * 2       // R vaut 2469.12
```

MANIPULATION DES DATES

En Windev, une date est enregistrée la plupart du temps sous forme d'une chaîne de caractères, au format *AAAAMMJJ*.

Pour convertir une date du format interne *AAAAMMJJ* au format traditionnel *JJ/MM/AAAA*, utiliser la fonction *DateVersChaine* :

```
MaDate est une chaine = "20070312"
Info(DateVersChaine(Madate, "JJ/MM/AAAA")) // Affiche 12/03/2007
```

La fonction *ChaineVersDate* a l'effet inverse.

Pour effectuer des calculs sur les dates, le mieux est de convertir la date en un entier.

```
MaDate est une chaine = "20070331"
D est un entier
D=DateVersEntier(Madate)
D++
Madate=EntierVersDate(D)
Info(DateVersChaine(Madate, "JJ/MM/AA")) // Affiche 01/04/07
```

On aurait pu aussi écrire de façon plus concise (mais peut être moins lisible) :

```
MaDate est une chaine = « 20070331 »
Info(DateVersChaine(EntierVersdate(DateVersEntier(Madate)+1),"JJ/M
M/AA")) // Affiche là aussi 01/04/07
```

Pour connaître la date du jour, utilisez la fonction *DateDuJour* ou *DateSys* :

```
MaDate est une chaine = DateDuJour()  // Date au format AAAAMMJJ
```

Notez dans l'exemple ci-dessus que tout appel à une fonction Windev doit être suivi de parenthèses juste après le nom, même si la fonction n'attend aucun paramètre.

Pour connaître le nombre de jours écoulés entre deux dates, utilisez la fonction *DateDifférence* :

```
MaDate est une chaine = "20070301"
Info(DateDifférence(Madate,DateSys()) // Affiche 27 si on est le
28/03/07
```

Autres fonctions utiles sur les dates :

```
MaDate est une chaine = "20070331"
Info(DateVersJour(madate))            //    Donne    5    (1=Lundi,
2=mardi…)
Info(DateVersJourEnLettre(madate))    // Jeudi
Info(DateVersMoisEnLettre(madate))    // Mars
```

GESTION DES CHAMPS

Les champs sont les éléments de base constituant les fenêtres (et les états) d'une application. A chaque type de champ correspond un « objet graphique » dessiné par Windows lors de l'affichage d'une fenêtre ou d'un état. Les principaux types de champ sont :

- Les champs de saisie
- Les libellés
- Les boutons
- Les sélecteurs (case à cocher) et interrupteurs (boutons radio)
- Les listes déroulantes (appelées aussi Combo)
- Les tables

Ces champs peuvent réagir à des évènements : un clic sur un bouton va par exemple déclencher un traitement ; la sortie d'un champ de saisie va déclencher un contrôle de la valeur saisie. Les évènements principaux pris en compte par ces champs sont :

- L'initialisation (réalisée une seule fois à l'ouverture de la fenêtre)
- L'entrée ou la sortie du champ (lorsque le focus arrive ou ressort du champ, par la touche *Tabulation* ou par un clic de souris dans la fenêtre)

- Un clic (sur un bouton par exemple)
- La frappe d'un caractère dans un champ de saisie
- La sélection d'une valeur dans une liste déroulante…

La fenêtre elle-même qui contient tous ces champs peut réagir à certains évènements : ouverture, fermeture, prise ou perte de focus, modification de taille…

Lors de l'écriture d'un Macro-Code Utilisateur, vous aurez bien souvent à « lier » votre code à l'un des évènements décrits ci-dessus, soit pour un champ précis de la fenêtre, soit pour la fenêtre elle-même.

Tous les champs peuvent être manipulés au travers de propriétés. Certaines propriétés sont spécifiques à un type de champ, d'autres sont génériques. La syntaxe générale pour accéder ou modifier la valeur d'une propriété est : *NomChamp..Propriété*

Les propriétés les plus courantes sont :

Visible	Etat de visibilité du champ : *Vrai* ou *Faux*
Etat	Etat du champ : *Actif*, *Grisé*, *AffichageSeulement* ou *Inactif*
Libellé	Libellé d'un champ de saisie, d'un bouton, d'une combo
X et *Y*	Position horizontale et verticale du champ dans la fenêtre
Message	Texte affiché dans la barre de message lorsque le focus est sur ce champ
Bulle	Texte affiché dans la bulle d'aide associée à un champ
Couleur	Couleur du texte affiché dans le champ
CouleurFond	Couleur de fond d'un champ
Hauteur,Largeur	Hauteur et Largeur d'un champ ou d'une fenêtre
Taille	Donne le nombre de caractères contenus dans un champ de saisie
Occurrence	Nombre de lignes présentes dans une liste ou une table
Titre	Titre d'une colonne de table, ou d'une fenêtre

Quelques exemples :

```
CST1..Libellé="Unité" // Modifie le libellé du champ de saisie
MODIF..Etat=Grisé   // Grise le bouton MODIF (qui n'est plus clicable)
CST1
MaDate est une chaine = "20070331"
```

Les couleurs peuvent être manipulées par des constantes prédéfinies dans le langage, ou par spécifications directe des trois composantes (Rouge, Vert, Bleu).

Couleurs prédéfnies : *iNoir, iBlanc, iBleuClair, iBleuFoncé, iRougeClair, iRougeFoncé*…

Spécification d'une couleur : *RVB(128,0,255)* // Chaque composante est comprise entre 0 et 255

Exemples :

```
UnChamp..Couleur = iBleuClair
Table1[5]..CouleurFond = RVB(128,128,128)   //     équivalent     à iGrisFoncé
```

COMMENT DIALOGUER AVEC L'UTILISATEUR

Windev permet de dialoguer avec l'utilisateur grâce à des boites de dialogue. On distingue plusieurs types de boites de dialogue :

- ⇨ Les boites d'information, d'avertissement ou d'erreur : ces boites affichent un message d'information ou d'erreur, mais aucun choix n'est possible. Il suffit de cliquer sur le seul bouton *OK* pour fermer la boite.
- ⇨ Les boites de question, ou vous devez répondre par *Oui* /*Non* ou *OK / Annuler* à la question posée.

⇨ Les boites de confirmation, ou trois réponses sont possibles *Oui*, *Non* ou *Annuler*.

Pour afficher une boite de dialogue, utilisez l'une des fonctions ci-dessous :

- ➙ *Info* Affiche un simple message d'information
- ➙ *Avertissement* Affiche un message d'avertissement
- ➙ *Erreur* Affiche un message d'erreur
- ➙ *OuiNon* Affiche une boite de question, avec réponse *Oui Non*
- ➙ *OKAnnuler* Affiche une boite de question, avec réponse *OK Annuler*
- ➙ *Confirmer* Affiche une boite de confirmation

Toutes ces fonctions reçoivent en paramètres le texte qui sera affiché dans la boite de dialogue. Ce texte peut être spécifié en plus parties.

Exemples :

```
 SI Code <> "001" et Code <> "002" alors
       Erreur("Seules les valeurs 001 et 002 sont autorisées.")
       DonneFocus(MonChamp)  // Pour reprendre la saisie sur le champ
ayant le nom MonChamp
       Renvoyer 0
 Fin


 SI pas OuiNon("Etes-vous sûr(e) de vouloir clôturer ?") alors
       Renvoyer 0  // Pour retourner en saisie
 Fin

Info("Première ligne de texte", "Deuxième ligne", ...
       "Troisième ligne de texte")
```

Les fonctions *OuiNon* et *Confirmer* peuvent recevoir, en premier paramètres, la valeur par défaut qui sera proposée comme réponse.

```
  // Dans l'exemple ci dessous, c'est la valeur Non
  // qui est proposée par défaut    (celle qui est prise si on fait
  Entrée)
  SI OuiNon(Non, "Etes-vous sûr(e)?") alors
```

La fonction *DélaiAvantFermeture*, utilisée juste avant l'affichage d'une boite de dialogue, permet de limiter le temps d'affichage de la boite. Au bout du délai imparti, si l'utilisateur n'a pas répondu à la question posée (ou n'a pas « acquitté » le message d'information ou d'erreur), la réponse par défaut est automatiquement sélectionnée, la boite de dialogue est refermée, et le traitement se poursuit.

```
  // Dans l'exemple ci dessous, c'est la valeur Non
  // qui est proposée par défaut    (celle qui est prise si on fait
  Entrée)
  DélaiAvantfermeture(500)     // On a 5 secondes pour répondre Non,
                               // Sinon, la réponse automatique sera
  Oui
  SI OuiNon("Confirmez-vous la validation de cette fiche ?") alors
```

La fonction *TitreSuivant* permet de personnaliser le titre de la prochaine boite de dialogue qui sera ouverte :

```
  TitreSuivant("Confirmation de suppression")
  SI OuiNon("Etes vous sûr(e) de vouloir supprimer cette rubrique ?")
```

Enfin, la fonction saisie, plus élaborée, permet d'ouvrir une boite de dialogue et de saisir une information.

```
// Exemple de traitement protégé par un mot de passe
MotPasse est une chaine
SELON Saisie("Indiquez le mot de passe pour ce traitement ?",
MotPasse)
   // OK
   CAS 1 : SI MotPasse <> "ABCDEF" alors Renvoyer 0
   // Annuler
   CAS 2 : Renvoyer 0
FIN
```

AUTRES FONCTIONS UTILES DU LANGAGE

La fonction *ToucheEnfoncée* permet de savoir à un instant t si une des touches *Majuscule* ou *Control* du clavier est enfoncée. Cela permet par exemple de déclencher un traitement particulier à l'ouverture ou la fermeture d'une fenêtre, lors d'un clic sur un bouton, uniquement en fonction de l'état de ces touches.

```
SI ToucheEnfoncée(teShift) Alors  // Si Majuscule est enfoncée
SI ToucheEnfoncée(teControl) Alors// Si Majuscule est enfoncée
```

Les touches pouvant être testées sont par exemple : *teShift, teControl, teAlt*

La fonction *Bip* permet d'émettre un bip sonore

La fonction *RéseauUtilisateur* retourne le nom de l'utilisateur Windows connecté sur le poste de travail. Cela peut s'avérer pratique pour sécuriser certaines fonctions de l'application.

```
SI RéseauUtilisateur()~= "Administrateur" alors
```

GESTION DES FICHIERS

Les fonctions d'accès à la base de données ne sont volontairement pas abordées, d'une part car il serait trop long de les décrire ici même de façon simplifiée, d'autre part car elles doivent être réservées à des utilisateurs avertis.

Sachez simplement que pour accéder, dans un morceau de code, à une rubrique d'un fichier, la syntaxe est *NomFichier.NomRubrique*.

Cette syntaxe peut être utilisée dès lors qu'un enregistrement du fichier a été lu auparavant. Ainsi, lors de l'écriture d'une fonction personnalisée, il est dit dans cette documentation que vous pouvez accéder aux rubriques des fichiers Personnel (*PEPERS*), Situations (*PEPACT*), et En-têtes Bulletin (*CAENBU*) (voir liste des rubriques dans l'annexe 1). Vous avez donc toute latitude pour écrire par exemple :

```
// Test Mode de paiement du salarié, et code statistique 1
SI PEPERS.MOPM= "VB" ET PEPACT.CST1= "002" alors
```

Pour aller plus loin dans la manipulation des fichiers, faites vous aider par un spécialiste Windev.

CONCLUSION

Tout ce qui a été décrit dans ce chapitre ne représente, vous l'aurez compris, qu'une toute petite part de ce qu'il est possible de réaliser avec le langage de Windev.

Cette documentation n'a pas prétention à être exhaustive ; elle a juste pour but de vous montrer que l'écriture de code Windev est à la portée d'un grand nombre d'utilisateurs. Elle s'avère d'ailleurs souvent plus facile que l'écriture de macros Excel !