

SILVERDEV

COMPOSANTS

Notice relative aux droits d'auteurs.

Les informations contenues dans ce document pourront faire l'objet de modifications sans préavis et ne sauraient en aucune manière engager **EXPERIA**. La fourniture du progiciel est régie par un octroi de licence ou un accord de confidentialité. Le progiciel ne peut être utilisé, copié ou reproduit sur quelque support que ce soit que conformément aux termes de cette licence ou de cet accord de confidentialité. L'acheteur ne peut effectuer des copies que dans le but de sauvegarde ou d'archivage.

Aucune partie du manuel et du progiciel ne peut être reproduite ou transmise par quelque moyen que ce soit, électronique ou mécanique, y compris par photocopie, enregistrement ou tout autre procédé de stockage, de traitement et de récupération d'informations, pour d'autres buts que l'usage personnel de l'acheteur sans permission expresse et écrite de la société **EXPERIA**.

IBM, AS/400, iSeries, System i, i5, Power I sont des marques déposées de International Business Machines Corporation. Windows est une marque déposée de Microsoft.

Tous les autres produits sont des marques déposées de leur société respective.

EXPERIA Europe
4, rue L.Beridot
Les jardins d'Epione
38500 VOIRON
FRANCE

Table des matières

Chapitre 1. Propriétés communes.....	5	Chapitre 13. CDateEdit.....	43
TWin	7	Chapitre 14. CCalendar	45
Chapitre 2. CMainMenu.....	8	Chapitre 15. CMaskEdit.....	46
Relier une fiche et un CMainMenu	8	EditMask	46
Editer un CmainMenu	8	Editeur spécifique	48
Séparateur	9	Chapitre 16. CImage.....	50
Images.....	9	Design time	50
Raccourcis	9	RunTime	50
Lettres soulignées	9	Zoomable.....	51
XpMenu	10	Chapitre 17. CSplitter	52
Chapitre 3. CPopupMenu.....	11	Sens	53
Relier un popupMenu à un composant.	11	Chapitre 18. CTreeView.....	55
Editer un popupMenu	11	Editer en design Time	55
Remarques	11	Editer en run Time	55
Chapitre 4. CMemo	12	Index.....	58
Utilisation	12	Interroger les éléments sélectionnés.	59
WordWrap.....	13	Cases à cocher	59
WantReturns	13	Copie en local	60
Text	14	Exemple de drag and Drop	60
Chapitre 5. CRichEdit	15	Chapitre 19. CTimer	62
Chapitre 6. CCheckBox.....	17	Chapitre 20. CTrayIcon	63
Chapitre 7. CRadioButton.....	18	Chapitre 21. CFontDialog, CColorDialog,	
Key et KeyChecked.....	18	COpenDialog, CSaveDialog,	
Chapitre 8. CComboBox	19	COpenPictureDialog,	
Items et Keys	19	CSavePictureDialog	65
Run Time	20	Chapitre 22. CNavBar.....	66
ComboBox trié	21	Conception dans le designer.....	66
Style	22	Ajout d'un control dans un groupe.	67
Chapitre 9. CListBox.....	23	OnLinkClick	67
Similarité avec CComboBox	23	Création dynamique de groupes.....	67
MultiSelect	23	Chapitre 23. CScheduler	68
Chapitre 10. CCheckListBox	25	Ajouter les événements	68
Chapitre 11. CActionList	26	Propriétés d'un évènement	68
Raccourci.....	26	Valeurs supplémentaires pour un évènement	69
Centralisation	27	DateNavigator	69
Chapitre 12. CSFL.....	28	Sélectionner une période par programme.....	70
Colonnes	28	Différentes vues	70
Exemple de boucle.....	29	Ressources	70
Lecture d'un SFL et lignes modifiées	30	Fiches et popupMenu internes	70
Formatage des cellules	31	Drag and drop d'un évènement	71
Column.style	33	Modifier la durée d'un évènement	72
ColumnField.DataType	35	Supprimer un évènement.....	73
Tabulation	36	Double clique sur un évènement	73
Tris	36	Double clique sur l'agenda.....	74
Impressions	37	Détecter le changement de période.....	74
Exports	39	Images dans un évènement.....	74
Multiselection	39	Couleur d'un évènement.....	75
Images.....	39	Programme exemple sddmview	75
Attributs d'affichage.....	41	Chapitre 24. CLinkLabel.....	76
		Link.Color :	76
		Link.Font :	76

Link.Url :	76	Propriétés	86
Chapitre 25. CCmdDialog	77	Legendes	86
Description.....	77	Etiquettes.....	87
Sélection de commande :	78	Titres.....	89
Onglet Texte	80	Axes.....	90
Paramètre de type commande.....	80	Pagination	95
Propriétés	81	Fond	96
Evènements :	81	Murs.....	98
CMapPoint	82	Zoom	99
Graphiques	82	3D 101	102
Chapitre 26. CChart	83	Couleurs	102
Chapitre 27. TChartSeries	84	Impression avec graphique	103
Fonctions	85	Pdf avec graphique.	103
		Gdi+	103
		Ascenseurs	104

Chapitre 1. Propriétés communes

Les propriétés suivantes se retrouvent sur la plupart des composants :

Align :

Utilisez Align pour aligner un contrôle en haut, en bas, à gauche ou à droite d'une fiche ou d'un volet de manière à ce qu'il reste à cet emplacement même si la taille de la fiche, du volet ou du composant contenant ce contrôle change. Lorsque le parent est redimensionné, un contrôle aligné modifie aussi ses dimensions afin de continuer à s'étendre vers le côté supérieur, inférieur, gauche ou droit du parent.

Par Exemple : , pour utiliser comme palette d'outils un composant volet contenant divers contrôles, attribuez la valeur `alLeft` à la propriété Align de ce volet. Cette valeur garantit le positionnement de la palette d'outils sur le bord gauche de la fiche avec une hauteur égale à celle de la hauteur client de la fiche.

La valeur par défaut de Align est `alNone`, ce qui signifie qu'un contrôle reste là où il a été placé dans la fiche ou le volet.

Anchors

Utilisez la propriété Anchors pour garantir qu'un contrôle maintient sa position actuelle par rapport à un bord de son parent, même si le parent est redimensionné. Lorsque son parent est redimensionné, le contrôle conserve sa position par rapport aux bords auxquels il est ancré.

Si un contrôle est ancré sur des bords opposés de son parent, le contrôle s'ajuste quand son parent est redimensionné. Par Exemple : , si i la propriété Anchors d'un contrôle est initialisée à `[akLeft,akRight]`, le contrôle se redimensionne quand la largeur de son parent change.

Anchors se redimensionne uniquement lorsque le parent est redimensionné. Ainsi, par Exemple : , si un contrôlé est ancré sur les côtés opposés d'une fiche à la conception et que la fiche est créée dans une taille maximale, le contrôle n'est pas étiré car la fiche n'est pas redimensionné après la création du contrôle.

caption :

Spécifie une chaîne de texte permettant à l'utilisateur d'identifier le contrôle.

constraints :

Utilisez la propriété Constraints pour spécifier les largeurs et hauteurs minimum et maximum du contrôle. Lorsque Constraints contient des valeurs maximum ou minimum, le contrôle ne peut pas être redimensionné pour violer ces contraintes.

Avertissement : Ne définissez pas de contraintes pouvant entrer en conflit avec la valeur de la propriété Align ou Anchors. Lorsque ces propriétés sont en conflit, la réponse du contrôle aux tentatives de redimensionnement n'est pas bien définie.

Cursor :

Modifiez la valeur Cursor pour indiquer à l'utilisateur quelle action est possible quand le pointeur de la souris entre dans le contrôle. La valeur de Cursor correspond à l'indice du pointeur dans la liste des pointeurs gérée par la variable globale Screen. Outre les pointeurs prédéfinis proposés par TScreen, une application peut ajouter des pointeurs personnalisés à la liste.

Font :

Détermine les attributs du texte écrit au-dessus ou dans le contrôle.

Height:

Indique la taille verticale du contrôle, exprimée en pixels.

Hint :

Contient la chaîne de texte apparaissant lorsque l'utilisateur déplace la souris au-dessus du contrôle.

Left :

Détermine la coordonnée horizontale, exprimée en pixels relativement à la fiche, du bord gauche d'un composant.

PopupMenu :

Identifie le menu surgissant associé au contrôle.

TabOrder :

Indique la position du contrôle dans l'ordre de tabulation de son parent.

TabStop :

Détermine si l'utilisateur peut tabuler dans un contrôle.

Tag :

Stocke une valeur entière dans un composant.

Top :

Représente la coordonnée verticale Y, exprimée en pixels relativement à son parent ou son contrôle conteneur, du coin supérieur gauche d'un contrôle.

Visible :

Détermine si le composant apparaît à l'écran.

Width :

Détermine la taille horizontale, exprimée en pixels, du contrôle ou de la fiche.

TWin

TWin représente une fenêtre standard d'une application.

BorderIcons :

Utilisez la propriété BorderIcons pour connaître ou définir les icônes qui apparaissent dans la barre de titre de la fiche. BorderIcons peut utiliser une combinaison quelconque des valeurs suivantes de TBorderIcons :

biSystemMenu La fiche a un menu Contrôle (également appelé menu Système).

biMinimize La fiche dispose d'un bouton Réduire.

biMaximize La fiche dispose d'un bouton Agrandir.

biHelp Si BorderStyle a pour valeur bsDialog ou si biMinimize et biMaximize sont exclues, un point d'interrogation apparaît dans la barre de titre de la fiche et, quand il est cliqué, le curseur devient crHelp ; sinon le point d'interrogation n'apparaît pas.

BorderStyle :

Utilisez la propriété BorderStyle pour connaître ou définir l'aspect et le comportement de la bordure de la fiche.

FormStyle :

Utilisez la propriété FormStyle pour connaître ou définir le style de la fiche. FormStyle peut prendre l'une des valeurs suivantes :

fsNormal La fiche n'est ni une fenêtre parent MDI ni une fenêtre enfant MDI.

fsMDIChild La fiche est une fenêtre enfant MDI.

fsMDIForm La fiche est une fenêtre parent MDI.

fsStayOnTop Cette fiche reste au-dessus du bureau et de toutes les fiches du projet, sauf celles dont la propriété FormStyle contient fsStayOnTop. Si une fiche à l'état fsStayOnTop en ouvre une autre, aucune des deux fiches ne reste systématiquement au-dessus.

WindowState :

Définissez la propriété WindowState pour réduire,agrandir ou restaurer la fenêtre de la fiche.Consultez WindowState pour déterminer si la fiche apparaît réduite,agrandie ou normale à l'écran.

Chapitre 2. CMainMenu

Onglet : Standard

Ce composant permet d'ajouter un menu principal à une fenêtre.

Relier une fiche et un CMainMenu

Les fiches ont une propriété nommée Menu.

Pour qu'une fiche utilise un CMainMenu, il faut que cette propriété Menu pointe sur le CMainMenu. Lorsque vous posez un CMainMenu sur une fiche, si la propriété Menu de la fiche est vide, elle prend automatiquement la valeur du CMainMenu posé.

Editer un CMainMenu

Pour éditer un CMainMenu dans le designer, il faut commencer par repérer le carré qui représente le CMainMenu. Ce représentant ne sera visible que dans le designer. Il ne sera pas visible lors de l'exécution du programme.

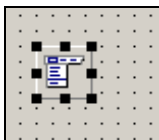


Figure 1

A partir du représentant, effectuez un click droit et choisissez le menu editeur (Figure 2).

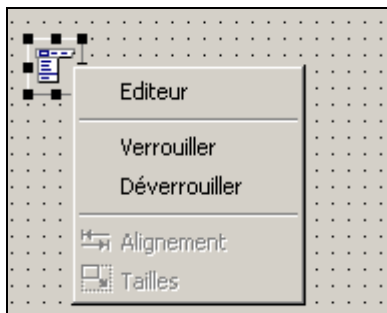


Figure 2

(Il est aussi possible d'effectuer un double click sur le représentant ou de choisir dans l'inspecteur de propriétés la propriété Items Figure 3)

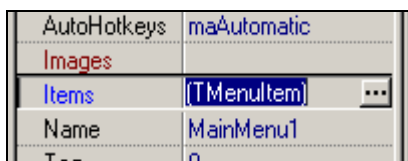


Figure 3

Un éditeur spécifique aux menus apparaît alors.

Cet éditeur permet d'ajouter des éléments de menu qui sont des objets de type TMenuItem. Pour ajouter un MenuItem, utilisez le click droit sur l'éditeur de menu, puis cliquez sur Nouveau.

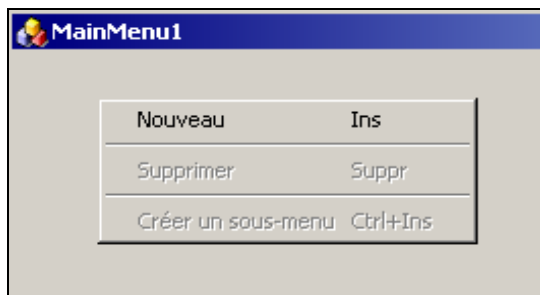


Figure 4

Le MenuItem apparaît à la fois sur le menu de la fiche et dans l'éditeur de menu. Pour modifier les propriétés du MenuItem, il suffit de le sélectionner dans l'éditeur de menu et de modifier les propriétés dans l'inspecteur de propriétés.

Séparateur

Pour ajouter un séparateur (Figure 5) dans un menu, il faut que la propriété caption de l'élément soit égale à un tiret.

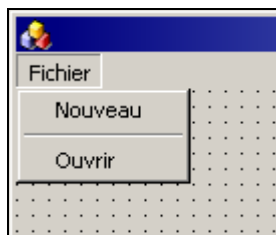


Figure 5

Images

La propriété Images permet de faire pointer le CMainMenu sur une liste d'images qui seront utilisées par les Items.

Il faut ensuite utiliser la propriété ImageIndex de l'élément de type TMenuItem.

Raccourcis

Pour créer un raccourci clavier, utilisez la propriété ShortCut de TMenuItem.

Lettres soulignées

Pour souligner un caractère dans le libellé, faites-le précéder d'un caractère &. Ce type de caractère est appelé caractère de raccourci. Si Caption comporte un caractère de raccourci, l'utilisateur peut sélectionner l'élément de menu en appuyant sur la touche Alt tout en appuyant sur le caractère souligné.

Selon le paramétrage du pc, le caractère sera souligné tout le temps ou seulement lorsque l'utilisateur appuie sur la touche ALT.

click droit sur le bureau : propriétés /apparences/effets/Masquer les lettres..

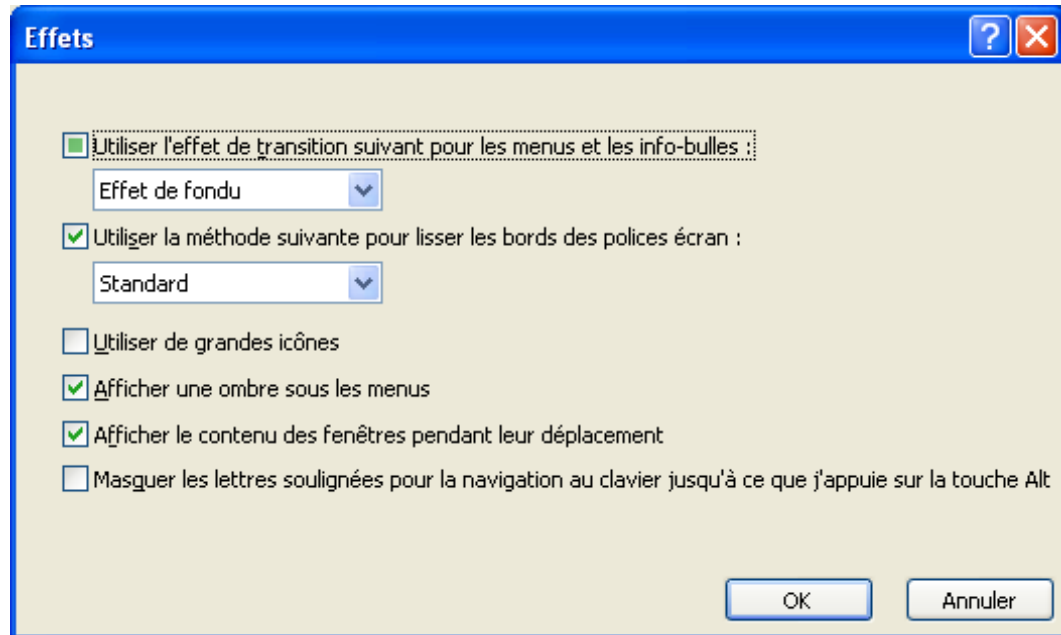


Figure 6

XpMenu

Il est possible de donner aux menus l'apparence d'un menu XP en ajoutant sur la fiche un composant Cxpmenu qui se trouve dans l'onglet système.(Il faut mettre la propriété Active du Xpmenu à true)

Chapitre 3. CPopupMenu

Onglet : Standard

Ce composant permet d'ajouter un menu surgissant.

Dans le designer, le popupMenu n'est pas directement visible. Il est représenté par un carré.

Ce représentant ne sera visible que dans le designer (Design Time). Il ne sera pas visible lors de l'exécution du programme (Run Time).

Relier un popupMenu à un composant.

Tous les composants dérivés de TControl possèdent une propriété PopupMenu.

Faites pointer cette propriété sur un PopupMenu de la fiche à l'aide de l'inspecteur de propriétés.(Figure 7)

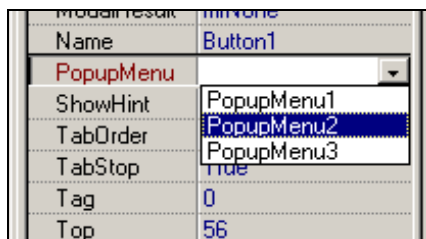


Figure 7

Editer un popupMenu

L'édition d'un PopupMenu dans le designer se fait de la même manière qu'un CMainMenu.

Remarques

Voir séparateur, images, raccourcis et XpMenu dans CMAinMenu

Chapitre 4. CMemo

Onglet : Standard

Utilisation

CMemo est un composant qui permet d'afficher du texte sur plusieurs lignes.

Le texte est dans un objet de type TStrings.

Vous retrouverez ce type d'objet dans plusieurs composants. Il s'agit d'un objet contenant des chaînes de caractères. Le nombre de chaînes de caractères contenues dans le Tstrings n'est pas limité

(ependant le composant CMemo est lui limité à 64 Ko Sous Windows 9x)

Les fonctions qui permettent de manipuler un objet de type TStrings sont :

sdSIAdd	Ajout d'une chaîne
sdSIClear	Effacement de toutes les chaînes
sdSIGet	Interrogation de la valeur d'une chaîne
sdSIInsert	Insert d'une chaîne

De plus , les objets de type TStrings possèdent une propriété Count qui donne le nombre de lignes.

Remarque :

Le premier élément de l'objet TStrings a l'indice 0.

L'exemple suivant montre comment remplir le composant à partir de la lecture d'un fichier et comment renseigner le fichier à partir de la lecture du composant.

Description du fichier :

```
R FCOMLIVRES
  IDLIVRE   P  4,0      1      3      3 N° LIVRE
  IDLINE    P  2,0      4      2      5 N° LIGNE
  LINE      A 100      6     100     105 LIGNE
K  IDLIVRE
K  IDLINE
```

Fichier vers composant :

```
P LOADCOMMENTS  B
D                PI
C                callp      sdSIClear (F2: 'Comments': 'Lines')
C  IDLIVRE       settl      FCOMLIVRES
C  IDLIVRE       reade (N)  FCOMLIVRES
```

77

```

C      *in77          doweq      *off
C                      callp      sdSlAdd(F2:'Comments': 'Lines':LINE)
C
C      IDLIVRE         reade(N)    FCOMLIVRES                                77
C                      enddo
P                      E

```

Composant vers fichier :

```

P WRITECOMMENTS      B
D                      PI
D Nbline              s              5u 0
D WIdLivre            s              like(IDLIVRE)
C                      eval          WIdlivre=IDLIVRE
C      WIDLIVRE        setll          SDDMCMLV
C      WIDLIVRE        READE          SDDMCMLV                                76
C      *in76           doweq          *off
C                      delete         SDDMCMLV
C      WIDLIVRE        READE          SDDMCMLV                                76
C                      enddo
C                      eval          NbLine =sdGetNum(F2:'Comments':
C                                     'Lines.Count')
C                      eval          IDLINE =0
C                      dow            IDLINE <  NbLine
C                      eval          LINE=sdSlGet(F2:'Comments.Lines'
C                                     :IDLINE)
C                      eval          IDLINE = IDLINE + 1
C                      WRITE          FCOMLIVRES
C                      enddo
P                      E

```

WordWrap

Affectez la valeur True à la propriété WordWrap pour que le contrôle de saisie fasse passer à la ligne le texte sur la marge droite afin qu'il tienne dans la zone client. Le passage à la ligne n'est qu'apparent, le texte n'inclut aucun caractère de passage à la ligne qui n'ait pas été explicitement saisi. Affectez la valeur False à la propriété WordWrap pour que le contrôle de saisie ne montre des changements de ligne que là où des caractères de passage à la ligne ont été explicitement saisis dans le texte.

Remarque : Il n'y a pas besoin d'une barre de défilement horizontale si WordWrap a la valeur True.

WantReturns

Détermine si l'utilisateur peut insérer des passages à la ligne dans le texte.

Affectez la valeur True à la propriété WantReturns pour permettre à l'utilisateur de saisir des caractères de passage à la ligne dans le texte. Affectez la valeur False à la propriété WantReturns pour permettre à la fiche de gérer les caractères de passage à la ligne.

Par exemple, dans une fiche ayant un bouton par défaut (un bouton OK) et un contrôle mémo, si `WantReturns` a la valeur `False`, l'appui de la touche Entrée choisit le bouton par défaut. Si `WantReturns` a la valeur `True`, l'appui de la touche Entrée insère un caractère de passage à la ligne dans le texte.

Remarque : Si `WantReturns` a la valeur `False`, les utilisateurs peuvent toujours insérer un passage à la ligne dans le texte en appuyant sur `Ctrl+Entrée`.

Text

Il est possible de récupérer le texte en une seule fois en utilisant la propriété `Text`. Entre chaque chaîne de caractères de l'objet `Lines`, deux caractères seront insérés : un retour chariot et un retour à la ligne.

Chapitre 5. CRichEdit

CRichEdit est un composant équivalent au composant CMemo avec la possibilité de formater le texte (selFontName, selFontStyle, selFontSize, selFontColor, selBgColor)

Exemple 1 :

```
C      callp      sdSetString(F1: 'RichEdit1': 'SelText':
C              'le ' : *OFF)
C      callp      sdSetSet(F1: 'RichEdit1': 'SelFontStyle':
C              'fsBold': 'fsUnderline')
C      callp      sdSetString(F1: 'RichEdit1': 'SelFontName':
C              'Arial')
C      callp      sdSetNum(F1: 'RichEdit1': 'SelFontSize':
C              15)
C      callp      sdSet(F1: 'RichEdit1': 'SelFontColor':
C              'clRed')
C      callp      sdSet(F1: 'RichEdit1': 'SelBgColor':
C              'clYellow')
C      callp      sdSetString(F1: 'RichEdit1': 'SelText':
C              'mot')
```

Résultat :



Figure 8

Exemple 2 :

Surligner un mot dans le texte :

```
Dmot      c      'peux'
Dlines    s      *
Dline     s      1000   varying
Dlentot   s      10u 0  inz(0)
Dpos      s      10i 0
D i       s      10u 0
C      eval      lines = sdgetlist(f1: 'richedit1.lines')
C      if        lines <> *null
C      for      i=0 to sdgetlistcount(lines)-1
C      eval      line=sdgetlistLabel(lines:i)
C      eval      pos= -1
```

```
C      eval      pos=%scan(mot:line)
C      dow      pos <>0
C      callp     sdsetint(f1:'richedit1':'selstart':
C                lentot + Pos-1)
C      callp     sdsetint(f1:'richedit1':'sellength':
C                %len(mot))
C      callp     sdSet(F1:'RichEdit1':'SelBgColor':
C                'clYellow')
C      if        pos + %len(mot) < %len(line)
C      eval      pos=%scan(mot:line:pos +%len(mot))
C      else
C      eval      pos=0
C      endif
C      enddo
C      eval      lentot=lentot+ %len(line) +2
C      endfor
C      callp     sdfreelist(lines)
C      endif
```

Chapitre 6. CCheckBox

Onglet : Standard

Le composant CCheckBox est une case à cocher. Il permet de représenter un booléen.

Lorsque le CheckBox est coché, la propriété Checked est à true.

Pour modifier ou interroger la valeur de Checked, deux méthodes sont possibles.

Utilisez les fonctions sdsetBool et sdGetBool sur la propriété Checked.

Exemple :

C	callp	sdSetBool (F1: 'CHK1' : 'CHECKED' : *OFF)
---	-------	---

Le composant possède aussi les propriétés Value, ValueFalse et ValueTrue.

Ces trois propriétés fonctionnent ensemble.

Ces trois propriétés sont de type chaîne de caractères. Il est donc possible de les modifier à l'aide des fonctions sdGet et sdSetString.

Lorsque vous interrogez la valeur de la propriété Value, vous récupérez la valeur de la propriété ValueFalse si la case est décochée et de ValueTrue si la case est cochée.

De même, si vous affectez la chaîne correspondant à ValueTrue dans Value, ceci cochera la case. (n'importe quel autre valeur décochera la case)

Chapitre 7. CRadioButton

Onglet : Standard

Utilisez CRadioButton pour ajouter un bouton radio à une fiche. Les boutons radio proposent à l'utilisateur un ensemble d'options mutuellement exclusives, c'est-à-dire qu'un seul bouton radio d'un groupe peut être sélectionné à la fois à un moment donné. Quand l'utilisateur sélectionne un bouton radio, le bouton radio précédemment sélectionné devient désélectionné. Les boutons radio sont fréquemment regroupés dans une boîte groupe (CGroupBox). Commencez par ajouter la boîte groupe à la fiche, puis choisissez les boutons radio dans la palette des composants et placez-les dans la boîte groupe.

Par défaut, tous les boutons radio placés dans le même contrôlé fenêtré conteneur (comme CRadioGroup ou CPanel) appartiennent au même groupe. Par exemple, deux **boutons radio** d'une fiche ne peuvent être sélectionnés simultanément que s'ils appartiennent à des conteneurs différents, par exemple deux boîtes groupe.

Key et KeyChecked

Chaque RadioButton peut contenir une valeur de type chaîne.

Pour retrouver la valeur de la propriété Key du composant CRadioButton dont la propriété Checked est true parmi les CRadioButton faisant parti du même conteneur, utilisez la propriété KeyChecked de l'un d'eux..

Chapitre 8. CComboBox

Onglet : Standard

Le composant ComboBox permet de choisir un élément dans une liste déroulante

Items et Keys

Dans de nombreux cas, il est intéressant d'afficher un libellé dans la liste déroulante, mais c'est une valeur associée qui vous intéresse.

Par exemple, vous voulez afficher la liste des auteurs dans la liste déroulante, mais c'est le code de l'auteur choisi par l'utilisateur qui vous intéresse.

Pour cela, le composant ComboBox a les propriétés Items et Keys.

Ces deux propriétés sont de type TStrings, c'est à dire qu'il s'agit d'une suite de chaînes alphanumériques.(On peut comparer cela à un tableau de chaînes)

La propriété Items contient la liste des libellés qui seront affichés dans la liste déroulante.

La propriété Keys contient la liste des clefs.

Le développeur doit veiller à ce que le nombre d'éléments dans la propriété Keys et dans la propriété Items correspondent.

De plus, un code et un libellé qui correspondent doivent être à la même position dans les objets Items et Keys.

Les objets Items et Keys peuvent être renseignés en design time en utilisant l'inspecteur de propriétés.

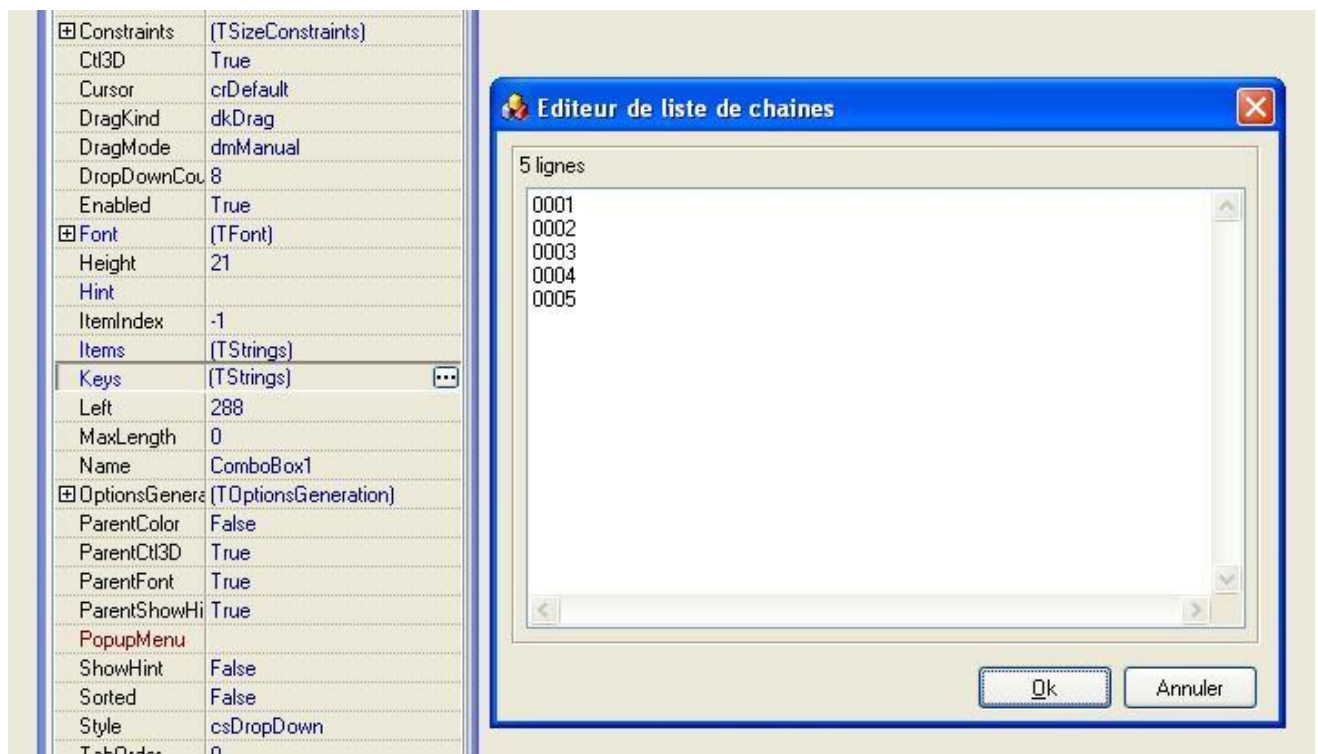


Figure 9

Exemple :

Chaînes contenues par Items	Chaînes contenues par Keys
Vernes jules	0001
Gary Romain	0002
Twain Mark	0003
Zola Emile	0004
Duras Marguerite	0005

L'utilisateur voit ceci :

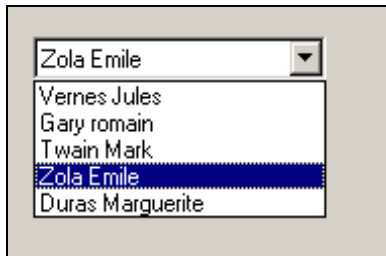


Figure 10

Mais ce qui intéresse le développeur, c'est de savoir quelle est le code de l'élément sélectionné. (Ici Emile Zola, de code 004)

Le développeur peut retrouver le code de l'élément sélectionné en interrogeant la propriété KeySelected du composant ComboBox.

```
C          eval          IDAUTEUR=sdGetInt (F1 : 'COMBOAUTEUR' :
C          'KeySelected')
```

Dans cet exemple :

F1 est l'identifiant de la fiche sur laquelle est posé le ComboBox.

COMBOAUTEUR est le nom du composant ComboBox et KeySelected est le nom de la propriété dont on veut récupérer la valeur dans la variable IDAUTEUR.

Remarque 1: Ici, on utilise la fonction sdgetInt car IDAUTEUR est numérique.

Si la clef est alphanumérique, il faut utiliser la fonction sdget.

Remarque 2: Le nom keys a été choisi car ce sont souvent les clefs d'une table qui y seront stockées, mais l'unicité des chaînes contenues dans Keys n'est pas obligatoire.

Run Time

Dans l'exemple précédent, la liste des libellés et des clefs était renseignée en design time.

Le plus souvent, les données contenues dans le comboBox proviennent d'une table et doivent donc être renseignées par programmation. Les objet de type TString (Items et Keys) se manipulent avec les fonctions :

sdAssignTo	Pour affecter un objet TString à un objet TString
sdSIAdd	Pour ajouter une chaîne dans un TString
sdSIClear	Pour Effacer un TString

sdSIGet	Pour récupérer la chaîne qui se trouve à une position dans le TStrings
sdSIInsert	Pour insérer une chaîne dans un TStrings

Exemple :

```

*Effacement des items et des keys
C          callp      sdSlClear(F1: 'COMBOAUTEUR': 'items')
C          callp      sdSlClear(F1: 'COMBOAUTEUR': 'keys')
C      *LOVAL      setll      AUTEURS
C          READ          AUTEURS                      55
C      *ON          DOWNE      *IN55
*Ajout à chaque enregistrement lu
C          callp      sdSlAdd(F1: 'COMBOAUTEUR': 'items':
C                      %trim(NOMAUT) )
C          callp      sdSlAdd(F1: 'COMBOAUTEUR': 'Keys':
C                      %char(IDAUT) )
C          READ          AUTEURS                      55
C          enddo

```

ComboBox trié

Le composant ComboBox possède une propriété Sorted de type Boolean qui détermine si les éléments de la liste sont triés par ordre alphabétique. Par défaut, cette propriété est à false.

Dans le cas ou elle est à true, les éléments sont triés selon les libellés. (Ce que voit l'utilisateur)

Les clefs sont alors déplacées automatiquement afin que la correspondance entre clefs et libellés soit sauvegardée.

Cependant lors du chargement, il faut remettre la propriété Sorted à false avant le chargement et la remettre à true après. Le code complet devient donc :

```

c          callp      sdSetBool(F1: 'COMBOAUTEUR': 'Sorted': *off)
*Effacement des items et des keys
C          callp      sdSlClear(F1: 'COMBOAUTEUR': 'items')
C          callp      sdSlClear(F1: 'COMBOAUTEUR': 'keys')
C      *LOVAL      setll      AUTEURS
C          READ          AUTEURS                      55
C      *ON          DOWNE      *IN55
*Ajout à chaque enregistrement lu
C          callp      sdSlAdd(F1: 'COMBOAUTEUR': 'items':
C                      %trim(NOMAUT) )
C          callp      sdSlAdd(F1: 'Liste': 'Keys':
C                      %char(IDAUT) )
C          READ          AUTEURS                      55
C          enddo
c          callp      sdSetBool(F1: 'COMBOAUTEUR': 'Sorted': *on)

```

Style

Si vous souhaitez que l'utilisateur ne puisse pas saisir dans la ComboBox, mettez la propriété `Style` à `csDropDownList`.

Chapitre 9. CListBox

Onglet : Standard

Ce composant permet d'afficher une liste de chaînes.

Son utilisation est très proche de celle du ComboBox.

L'intérêt de ce composant, est qu'il est possible de sélectionner plusieurs éléments (en utilisant les touches maj ou ctrl).

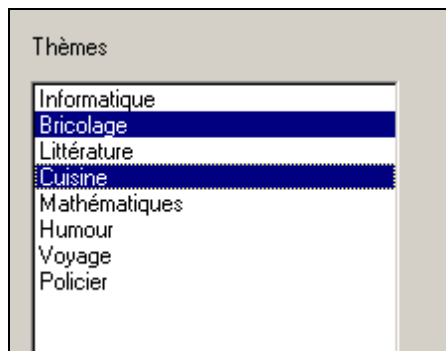


Figure 11

Similarité avec CComboBox

Pour l'utilisation des propriétés Items et Keys, (voir CcomboBox Run Time)

MultiSelect

Le composant CListBox possède une propriété MultiSelect de type boolean.

Cette propriété est par défaut à false. Lorsqu'elle est à true, l'utilisateur peut sélectionner plusieurs éléments.

Une propriété KeySelected existe dans CListBox. Cette propriété s'utilise comme pour le CComboBox, mais se révèle insuffisante lorsque la propriété MultiSelect est à true.

Dans ce cas, le développeur doit utiliser les propriétés FirstSelected et NextSelected.

Ces deux propriétés s'utilisent de concert pour connaître les éléments sélectionnés.

La propriété FirstSelected renvoie la position du premier élément sélectionné. (Le premier élément de la liste ayant l'indice 0)

Si aucun élément n'est sélectionné, cette propriété renvoie -1.

Ensuite, le développeur peut connaître la clef associée en utilisant la fonction sdSIGet sur la propriété keys.

De même, l'interrogation de la propriété NextSelected renvoi la position de l'élément suivant sélectionné ou -1 s'il n'y a plus d'élément sélectionné.

Exemple :

```
C                                     eval      Ligne=sdgetInt(F1: 'MultiSelect1':
```

```
C      'FirstSelected')
C      dow      ligne <> -1
C      eval      Clef = sdSlGet(F1:
C      'MultiSelect1': 'Keys': ligne)
C      eval      Ligne=sdgetInt(F1: 'Multiselect1':
C      'NextSelected')
C      enddo
```

Remarque :

Les propriétés FirstSelected et NextSelected sont en lecture seule.

Chapitre 10. CCheckListBox

Onglet : Standard

Ce composant est proche du composant CListBox.

Pour l'utilisation des propriétés Items et Keys, (voir CComboBox Run Time)

A côté de chaque chaîne de caractères, une case à cocher est dessinée. Il est possible de connaître la liste des cases cochées en utilisant les propriétés FirstChecked et NextChecked.

Exemple :

```
C      eval      Ligne=sdgetInt(F1: 'CheckSelect1':  
C      'FirstChecked')  
C      dow      ligne <> -1  
C      eval      Clef = sdSlGet(F1:  
C      'CheckSelect1': 'Keys': ligne)  
C      eval      Ligne=sdgetInt(F1: 'CheckSelect1':  
C      'NextChecked')  
C      enddo
```

Chapitre 11. CActionList

Onglet : Standard

Le composant CActionList est un composant non visuel.

L'objet CActionList en lui même n'a pas d'intérêt.

En revanche, ce composant possède une collection d'objets de type TAction qui peuvent être utiles.

Pour accéder aux objets TAction en Design Time, double cliquez sur l'icône représentant l'objet CActionList. Vous accédez alors à l'éditeur de collection.



Figure 12

Il vous est alors possible de modifier les propriétés de chaque élément de la collection par l'inspecteur de propriétés.

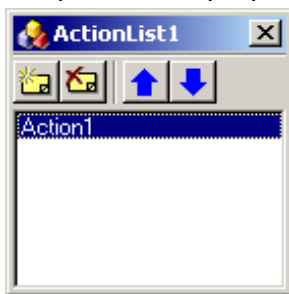


Figure 13

Raccourci

L'intérêt principal de l'objet TAction est qu'il possède une propriété ShortCut.

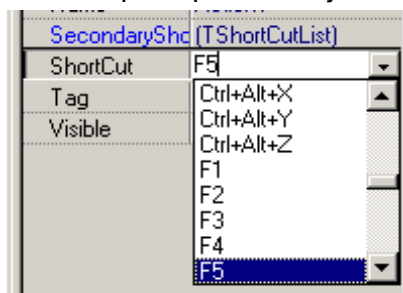
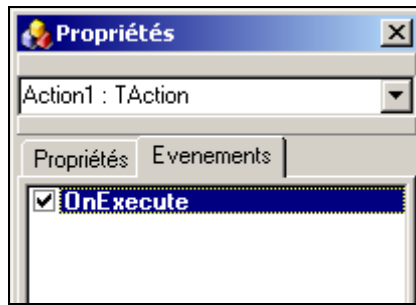


Figure 14

Si vous affectez la valeur F5 à la propriété ShortCut d'un TAction, l'objet TAction déclenchera un événement.

Demandez le signalement de cet événement dans l'inspecteur de propriétés en utilisant l'onglet événements.

**Figure 15**

Il est ainsi possible de déclencher une fonction RPG lorsque l'utilisateur appuiera sur la touche F5.

Remarque :

Attention aux conflits de raccourcis.

Centralisation

Le second intérêt des objets de type TAction est que de nombreux composants peuvent pointer sur un TAction. (Bouton, MenuItem ...)

Lorsqu'un composant pointe sur un TAction, un des ses événements (souvent OnClick) déclenche l'évènement OnExecute du TAction

Exemple :

Vous souhaitez que le click sur un bouton et le click sur un élément de menu déclenchent la même procédure sur le serveur.

Dans ce cas, il n'est pas nécessaire de demander le signalement des événements sur le bouton et l'élément de menu.

Il suffit de faire pointer les propriétés «action» du bouton et de l'élément de menu vers un objet TAction et d'associer une fonction à l'évènement OnExecute du TAction.

De même, la modification (en design time ou en run time) d'une propriété du TAction entraîne la modification de la même propriété sur tous les composants pointant sur ce TAction.

Chapitre 12. CSFL

Onglet : Supplément

Le composant CSFL permet d'afficher des données dans une grille.

Les données peuvent être modifiables ou non.

Chaque colonne peut avoir des propriétés d'affichage bien précises.

Ce composant est donc l'équivalent des sous fichiers sur iSeries/400, d'où son nom. (SFL pour Sub File).

Colonnes

Lorsque vous déposez un composant CSFL dans le designer, il n'a aucune colonne.

Pour ajouter des colonnes, il faut passer par l'éditeur de collection. (Les colonnes sont une collection d'objets du type TSFLColumn).

Pour accéder à l'éditeur de collection, éditez la propriété Columns dans l'inspecteur de propriétés.

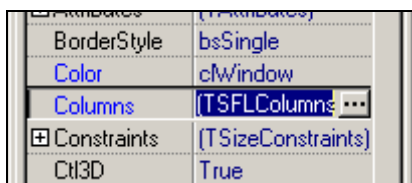


Figure 16

Il est aussi possible d'accéder à l'éditeur de collections en double cliquant sur le composant. Dans tous les cas, vous arriverez à la fenêtre suivante.



Figure 17

Utilisez les deux boutons pour ajouter ou supprimer des colonnes.

Chaque colonne a ses propres propriétés.

Les plus importantes sont Style et ColumnField.FieldName.

la propriété `style` permet de déterminer le type d'éditeur in situ de la colonne. (Editeur de date, d'heure, de booléen, une liste déroulante, un bouton qui déclenche un événement ou une simple saisie)

La propriété `ColumnField.FieldName` est une chaîne de caractères qui permet de renseigner la valeur d'une cellule. En effet, il serait possible de modifier la valeur d'une cellule avec les fonctions classiques par le code suivant :

```
C      callp sdSetString(f1:'sf11':'cells(4,5)':'Démon')
```

Ce qui inscrirait le texte 'Démon' dans la cellule ligne 5, colonne numéro 4.

Mais cette méthode est déconseillée car les colonnes peuvent être déplacées par l'utilisateur et vous ne saurez plus à quoi correspond la colonne 4.

De plus elle est peu pratique car dans une boucle qui renseigne toutes les lignes, il faudrait convertir un compteur numérique en alphanumérique de la manière suivante :

```
C      callp sdSetString(f1:'sf11':'cells(4,'+%char(cpt)+'')':'Démon')
```

Pour éviter ce problème, des fonctions spéciales du composant CSFL existent. Il s'agit des fonctions commençant par `sdSetCell`.

Ces fonctions prennent comme paramètre le nom `columnField.FieldName` et le numéro de ligne.

Exemple :

```
C      callp sdSetCell(f1:'sf11':'NomColonne':cpt:'Démon')
```

Il existe différentes fonctions qui prennent en dernier argument une chaîne de caractères, ou une variable numérique, ou une variable de type date etc...

Ces fonctions sont :

`sdSetCell`, `sdSetCellNum`, `sdSetCellDate`, `sdSetCellTime`, `sdSetCellNull`

Ces fonctions ont aussi un avantage. Il n'est pas nécessaire que les ligne existent dans le CSFL avant de les renseigner.

Le fait d'appeler une fonction de ce type sur la ligne 10 alors qu'il n'y a que 5 lignes dans le composant va automatiquement ajouter le nombre nécessaire de lignes.

Remarque :

Dans l'expression `cells(4,5)`, 4 correspond à la cinquième colonne car les colonnes commencent à 0 et 5 correspond à la sixième ligne. En général la ligne d'entête correspond à la ligne 0.

Exemple de boucle

Voici un exemple qui montre comment renseigner par code le contenu d'une grille.

La grille comporte deux colonnes. Une contenant l'identifiant d'un livre et l'autre son titre.

Remarque : Il est tout à fait possible de rendre invisible la colonne contenant l'identifiant.

```

C      callp      sdClearSubFile(MainForm: 'SubFile1')
C      eval      Row = 0
C      *LOVAL     setll      sddmliv
C      READ      sddmliv      55
C      *ON        DOWNE     *IN55
C      eval      Row =Row + 1
C      callp      sdSetCellNum(MainForm: 'SubFile1': 'IDLIVRE':
C                  Row:L_IDLIVRE)
C      callp      sdSetCell(MainForm: 'SubFile1': 'TITRE':
C                  Row:L_TITRE)
C      READ      sddmliv      55
C      ENDDO
C      callp      sdReinitAll(MainForm: 'SubFile1')
```

Lecture d'un SFL et lignes modifiées

Si la grille est en saisie, il faut alors relire la grille pour remettre la base de données à jour. Il n'est pas nécessaire de relire chaque ligne. Le composant CSFL possède des propriétés FirstModified et NextModified qui permettent de parcourir les lignes modifiées. Ensuite, il faut lire le contenu des cellules à l'aide des fonctions de type sdgetCell.

Lorsqu'une ligne est modifiée par l'utilisateur ou par programmation, elle est marquée comme modifiée. Pour qu'elle ne soit plus marquée comme modifiée, il faut utiliser la fonction sdReinit qui prend en argument le numéro de la ligne.

Pour marquer toutes les lignes comme non modifiées, il faut utiliser la fonction sdReinitAll. C'est pourquoi dans l'exemple précédent, un appel à cette fonction suit la boucle de remplissage.

Les propriétés FirstModified et NextModified renvoient -1 s'il n'y a plus de ligne modifiée.

Exemple :

```

PVALIDATION      B
D      PI
D form      5u 0 value
DRow      s      10 0
*****
C      eval      Row = sdGetInt(f1:'SFL1': 'FirstModified')
C      dow      Row <> -1
C      eval      test =sdGetCellNum(f1:'SFL1': 'IDLIVRE':Row)
C      test      chain      LIVRES      55
C      if      *IN55 =*OFF
C      eval      L_TITRE =sdGetCell(f1:' SFL1': 'TITRE':Row)
C      endif
C      eval      Row = sdGetInt(f1:'SFL1': 'NextModified')
```

```

C          enddo
C          callp      sdReinit(f1: 'SFL1')

```

Remarque :

Il est possible de visualiser les lignes modifiées. Pour cela, incluez les valeurs `goIndicator` et `goStar` dans la propriété `Options`. La valeur `goIndicator` indique qu'une colonne fixe est ajoutée à gauche et la valeur `goStar` indique qu'une étoile apparaîtra dans cette colonne pour chaque ligne modifiée.

Remarque :

*Il existe une autre méthode pour récupérer le contenu d'un sous fichier. La fonction `sdGetSfl` permet de recopier en local les données affichées dans une grille. Voir **Erreur ! Source du renvoi introuvable.***

Formatage des cellules

Dans une cellule, vous pouvez modifier la couleur du texte, la couleur du fond, le style de la police, l'alignement ou bien mettre une image de fond.

Le composant CSFL utilise en premier sa propriété `Color` pour déterminer le fond des cellules.

Ensuite, la propriété `Attributes` est utilisée pour déterminer le formatage des cellules paires ou impaires.

Puis la propriété `ColumnAttribute` des colonnes est utilisée pour déterminer le formatage des cellules de cette colonne.

Enfin, par programmation, il est encore possible de modifier la couleur du fond de la cellule en utilisant la propriété `ExAttributes`.

Exemple :

L'exemple suivant illustre la priorité entre les différents formatages.

	col1	col2

Figure 18

La propriété `color` du CSFL est `clWhite`

	col1	col2

Figure 19

La propriété Color du CSFL est clWhite
La propriété Attributes.Even.BgColor est clInfoBk

	col1	col2

Figure 20

La propriété Color du CSFL est clWhite
La propriété Attributes.Even.BgColor est clInfoBk
La propriété ColumnAttribute.BgColor de la colonne col1 est clLime

	col1	col2

Figure 21

La propriété Color du CSFL est clWhite
La propriété Attributes.Even.BgColor est clInfoBk
La propriété ColumnAttribute.BgColor de la colonne col1 est clLime
La propriété ExAttributes du CSFL possède un élément dont la propriété BgColor est clFuschia. Cet attribut a été appliqué par l'instruction :
`callp sdFormatCell(f1:'sf11':col1':3:0)`

Remarque : Pour les colonnes fixes, des couleurs permettant de donner l'impression de relief, sont automatiquement calculées.



Figure 22

Column.style

Les propriétés Column.style et ColumnField.DataType sont liées.
Modifier l'une des deux modifie la seconde.

La propriété style des colonnes peut prendre les valeurs suivantes :

csSimple :

Les cellules de la colonne sont éditables directement.

csButton :

Un bouton apparaît dans la cellule lorsque l'utilisateur clique.

King	20
King	10
King	10

Un évènement est déclenché lorsque l'utilisateur clique sur ce bouton. L'évènement onEllipsis.

Cet évènement possède des paramètres qui permettent de connaître la cellule où a eu lieu le click.

```
*/EVENT SFL1_OnEllipsis
d parm          ds          based(pevtfnf)
d Win           5u 0
d evt           48a
d ColName       100a  varying
d Row           10i 0
```

csPicklist :

Une liste déroulante apparaît lorsque l'utilisateur clique dans la cellule.

La colonne possède des propriétés PickItems et PickKeys.

Ces deux propriétés ne sont prises en compte que si la propriété style vaut csPickListe.

Dans ce cas, la cellule contiendra un éditeur in situ qui sera une liste déroulante.

Les éléments affichés dans la liste déroulante seront les chaînes de l'objet PickItems.

Exemple :

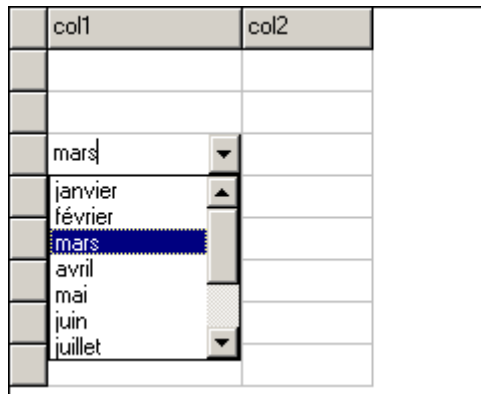


Figure 23

Si les données à afficher dans la liste déroulante proviennent de la base de données, il faut modifier les propriétés PickItems et PickKeys à l'exécution.

Pour cela, utilisez les fonctions sdColAddKey, sdColAddList, sdColClearKeys, sdColClearList

Exemple :

```

Pload          B
D              PI
DRow           s          10  0
  *Chargement de la liste déroulante pour les Thèmes
C              callp      sdColClearKeys(f1:'sf11':
C                      'NOMTHEME')
C              callp      sdColClearList(f1:' sf11':
C                      'NOMTHEME')
C      *LOVAL      setl1    THEMES
C              READ      THEMES                      55
C      *ON         DOWNE   *IN55
C              callp      sdColAddKey(f1:' sf11':'NOMTHEME':
C                      sdINTTOSTR(T_IDTHEME))
C              callp      sdColAddList(f1:' sf11':'NOMTHEME'
C                      %trim(T_NOMTHEME))
C              READ      THEMES                      55
C              enddo
P              E
  
```

csDate

L'utilisateur peut saisir une date directement ou cliquer sur un bouton et faire apparaître un calendrier.



Figure 24

csTime

L'utilisateur peut saisir directement une heure ou s'aider des boutons.

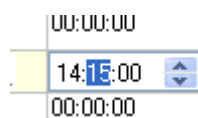


Figure 25

csBoolean

Des cases à cocher apparaissent dans les cellules.

Pour modifier par programme ces cellules, utilisez les valeurs de `columnField.PropBoolean.valueTrue` et `columnField.PropBoolean.valueFalse`

csImage

Une image est affichée dans la cellule. L'image affichée est prise dans l'imageList pointée par la propriété `images` du composant CSFL. Pour afficher l'image numéro `x`, écrivez la valeur `x` dans la cellule.

ColumnField.DataType

En fonction de la valeur de DataType et de style, les propriétés propAlpha, propBoolean, propNumeric, propDate et propTime sont prises en compte.

Exemple : si la propriété DataType est numeric, alors la propriété propNumeric est prise en compte.

Tabulation

Lorsque l'utilisateur utilise la tabulation dans un composant CSFL, le curseur se déplace de colonnes en colonnes puis à la ligne suivante.

La propriété TabStop à false permet d'indiquer que le curseur ne doit pas aller sur cette colonne.

Tris

Le click sur l'entête d'une colonne permet de trier le sous fichier selon cette colonne.

La propriété TypeSort permet d'indiquer si le tri est sensible à la casse ou non.

Il est possible de lancer un tri par programmation à l'aide de la fonction sdSort.

Prototype :

```
d sdSort          pr
d pform           5u 0 const
d SFL             30    varying value
d Column1        30    varying value
d Order1         10i 0 value
d Column2        30    varying value options(*nopass)
d Order2         10i 0 value          options(*nopass)
d Column3        30    varying value options(*nopass)
d Order3         10i 0 value          options(*nopass)
```

Paramètres :

SFL : Nom de la grille à trier

Column1 : nom de la colonne de tri

Order1 : Sens du tri : 1= ascendant , 2=descendant

Les paramètres suivants sont optionnels et permettent d'effectuer un tri secondaire.

Exemple :

C	callp	sdSort(F1:'sfl1':'Col1':1)
---	-------	----------------------------

La propriété sortable d'une colonne permet de déterminer si une colonne est automatiquement triée lors d'un click sur une colonne.

En affectant sortable à false et en utilisant l'événement onHeaderClick et la fonction sdSort, vous pouvez effectuer un tri personnalisé (avec des tris secondaire par exemple)

Exemple (en local) :

```
procedure sfl1_OnHeaderClick (Sender: TObject; ColName: String; Row: Integer);
begin
  if(colName = 'DATE') or (colName = 'HEURE') then
  begin
    _1.sfl1.sort('DATE',2,'HEURE',2,'',1);
  end;
end;
```

Impressions

Pour une impression formatée, utilisez les composants de la palette report.

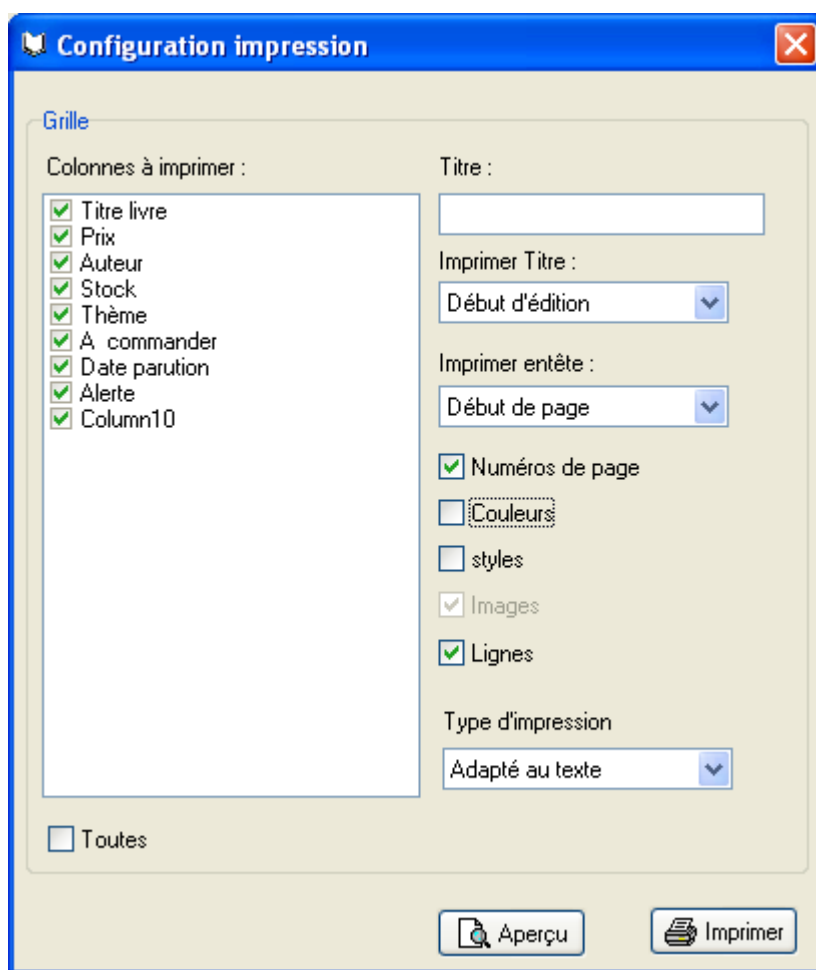
Cependant, il est possible d'imprimer le contenu d'un composant CSFL avec un minimum de code en utilisant la fonction sdPrint

C	callp	sdPrint(F1:'SFL1')
---	-------	--------------------

La propriété objet PrintParameters permet de paramétrer l'impression.

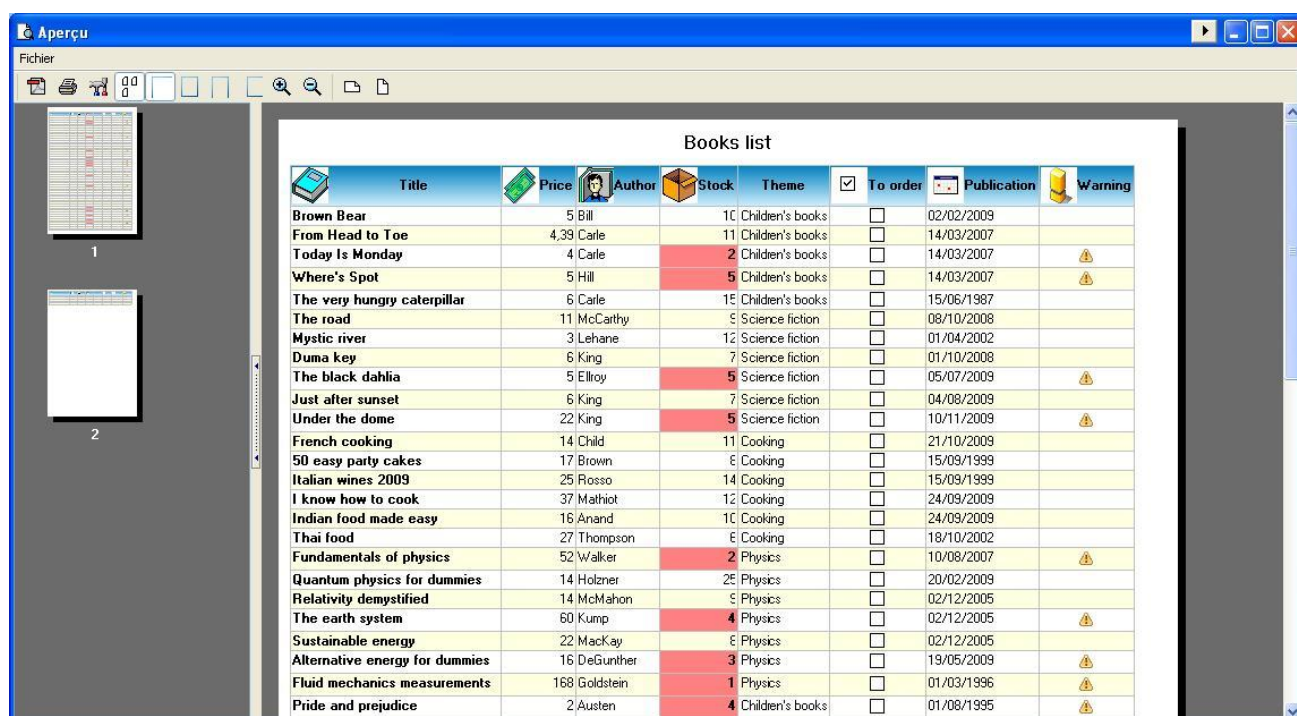
Chaque colonne possède une propriété printable afin d'indiquer les colonnes qui seront imprimées.

Si la propriété printParameters.showDialog est à true, une boîte de dialogue permet à l'utilisateur de modifier des paramètres de l'impression.

**Figure 26**

Enfin, la propriété `PrintParameters.UserCanChanges` permet au développeur de verrouiller des éléments de cette boîte de dialogue.

Il est aussi possible d'obtenir un aperçu avant impression en utilisant la fonction `sdPreview`.



Books list

Title	Price	Author	Stock	Theme	To order	Publication	Warning
Brown Bear	5	Bill	10	Children's books	<input type="checkbox"/>	02/02/2009	
From Head to Toe	4.39	Carle	11	Children's books	<input type="checkbox"/>	14/03/2007	
Today Is Monday	4	Carle	2	Children's books	<input type="checkbox"/>	14/03/2007	
Where's Spot	5	Hill	5	Children's books	<input type="checkbox"/>	14/03/2007	
The very hungry caterpillar	6	Carle	15	Children's books	<input type="checkbox"/>	15/06/1987	
The road	11	McCarthy	5	Science fiction	<input type="checkbox"/>	08/10/2008	
Mystic river	3	Lehane	12	Science fiction	<input type="checkbox"/>	01/04/2002	
Duma key	6	King	7	Science fiction	<input type="checkbox"/>	01/10/2008	
The black dahlia	5	Elroy	5	Science fiction	<input type="checkbox"/>	05/07/2009	
Just after sunset	6	King	7	Science fiction	<input type="checkbox"/>	04/08/2009	
Under the dome	22	King	5	Science fiction	<input type="checkbox"/>	10/11/2009	
French cooking	14	Child	11	Cooking	<input type="checkbox"/>	21/10/2009	
50 easy party cakes	17	Brown	8	Cooking	<input type="checkbox"/>	15/09/1999	
Italian wines 2009	25	Rosso	14	Cooking	<input type="checkbox"/>	15/09/1999	
I know how to cook	37	Mathiot	12	Cooking	<input type="checkbox"/>	24/09/2009	
Indian food made easy	16	Anand	10	Cooking	<input type="checkbox"/>	24/09/2009	
Thai food	27	Thompson	8	Cooking	<input type="checkbox"/>	18/10/2002	
Fundamentals of physics	52	Walker	2	Physics	<input type="checkbox"/>	10/08/2007	
Quantum physics for dummies	14	Holzer	25	Physics	<input type="checkbox"/>	20/02/2009	
Relativity demystified	14	McMahon	5	Physics	<input type="checkbox"/>	02/12/2005	
The earth system	60	Kump	4	Physics	<input type="checkbox"/>	02/12/2005	
Sustainable energy	22	MacKay	8	Physics	<input type="checkbox"/>	02/12/2005	
Alternative energy for dummies	16	DeGunther	3	Physics	<input type="checkbox"/>	19/05/2009	
Fluid mechanics measurements	168	Goldstein	1	Physics	<input type="checkbox"/>	01/03/1996	
Pride and prejudice	2	Austen	4	Children's books	<input type="checkbox"/>	01/08/1995	

Figure 27

Exports

Pour exporter le contenu d'un composant CSFL, utilisez la fonction `sdExport`.

Il y a trois types d'exportation possibles selon la propriété `TypeExport`

exBin Les données seront exportées dans un format binaire que excel pourra ouvrir. Excel n'a pas besoin d'être installé sur le poste

exCom Les données seront exportées au format excel en pilotant excel en com. Excel doit être installé sur le poste.

exCsv Les données seront exportées au format csv

Multiselection

En ajoutant l'elt `goMultiselect` dans la propriété `options`, l'utilisateur peut sélectionner plusieurs ligne. La touche `control` maintenu permet de sélectionner plusieurs lignes non adjacentes. La touche `shift` permet de sélectionner plusieurs lignes adjacentes.

L'utilisateur peut désélectionner des lignes en re cliquant dessus.

Images

Pour afficher une image dans un sous fichier, il faut tout d'abord mettre cette image dans une imagelist

il faut affecter cette imagelist à la propriété `images` du csfl

Ensuite il y a deux solutions :

1) Créer une colonne de type image (propriété style de la colonne = cslImage)

Pour afficher l'image numéro x de l'imagelist, il faut mettre la valeur x dans la cellule . -1 pour ne mettre aucune image)

2) Mettre l'image dans n'importequelle cellule (donc en plus du texte)

pour cela il faut créer un élément dans la collection exAttributes et affecter cet élément à la cellule avec la fonction sdFormatCell

TAttribute : TAttribute est un type possédant plusieurs propriété influant sur l'image.

Plusieurs sous propriété du composant CSL possèdent des propriétés du type TAttribute. Les propriétés de TAttribute influant sur l'image sont :

Images : La liste d'image dans laquelle sera prise l'image.

ImageIndex : L'indice de l'image dans la liste d'images ci dessus.

ImageStretched : La propriété ImageStretched de l'objet TAttribute indique si l'image doit s'étirer lorsque la cellule est agrandie.

Lorsque cette propriété est à false, l'image est en général plus jolie.

TransparentColor : La propriété TransparentColor de l'objet TAttribute indique quelle couleur doit être transparente. Si cette propriété a la valeur clNone, aucune couleur n'est transparente.

Layout : L'alignement vertical du texte et de l'image dépendent de la propriété Layout.

Priorité des objets TAttribute :

Pour savoir par exemple quelle liste d'images s'appliquent à une cellule, la liste d'image est recherché comme suit :

_Si la fonction sdFormatCell a été appelée, c'est la propriété Images du ExAttribute appliquée qui est utilisée.

_Si la fonction sdFormatCell n'a pas été appelée ou si la propriété Images du ExAttribute n'est pas renseignée, c'est la propriété images de la propriété ColumnAttribute de la colonne qui est appliquée (TitleAttribute si la cellule est en entête)

_Ensuite c'est dans la propriété images de la colonne que la liste d'image est recherchée

_Puis dans Attributes.Odd et Attributes.eve du sous fichier

_Puis dans la propriété images du sous fichier

Remarque : les propriétés Images du sous fichier et de la colonne ne sont plus utiles, mais ils sont conservé pour des raisons de compatibilité. En effet, l'objet TAttribute n'avait pas de propriété ImageList dans les premières version.

Remarque : Voir Formatage des cellules

Attributs d'affichage

Les attributs d'affichages (font, couleur de font, alignement vertical, alignement horizontale, image, image transparente, image étirée...) peuvent être modifiés à travers des objets de type Tattribute.

[-] TitleAttribute	[TAttribute]
Alignment	saCenter
BgColor	clBtnFace
FontColor	clNone
[+] FontStyle	[]
ImageIndex	0
Images	ImageList3
ImageStretche	False
Layout	slCenter
ReadOnly	False
TransparentCo	clNone

Figure 28

Plusieurs éléments possèdent des objets de type Tattribute.

L'attribut qui s'applique à une cellule suit les règles de priorité suivante (de la priorité la plus importante à la moins importante):

Attributs dû à un sdFormatCell (ExAttributes)

Attributs dû à la colonne (ColumnAttribute, Title.TitleAttribute)

Attributs dû à la parité de la ligne (Attributes.Even, Attributes.Odd)

Autres

Exemple :

La couleur de fond d'une cellule est celle de la propriété Color de CSubFile.

Ensuite , pour les lignes impaires, si la propriété odd.BgColor est différente de clNone, la cellule prend la couleur de la propriété odd.bgColor.

Ensuite, si la propriété ColumnAttribute.BgColor est différente de ClNone le fond de la cellule prend la couleur de la propriété ColumnAttribute.BgColor.

Ensuite, par programmation, il est encore possible de modifier la couleur du fond de la cellule en utilisant la propriété `SpecialsAttributes`.

Chapitre 13. CDateEdit

Onglet : Supplément

Ce composant permet de saisir une date.

Pour modifier ou interroger la valeur de la date représentée dans la zone d'édition, utilisez les fonctions `sdSetDate` et `sdGetDate` ou bien utilisez la propriété `Datelso`.

Un bouton est inclus dans la zone de saisie.

Lorsque l'utilisateur clique sur cette zone, une fenêtre surgit. Cette fenêtre comprend un composant `CCalendar`.

Pour modifier les propriétés du calendrier surgissant, utilisez les propriétés dont le nom commence par `Popup`. (`PopupAlign`, `PopupAutoClose`, `PopupBackGroundImg...`)

Les propriétés importantes de `CDateEdit` sont :

DirectInput : Détermine si l'utilisateur peut saisir directement une date ou s'il doit obligatoirement utiliser le calendrier.

PopupAutoClose : Détermine si le calendrier se referme automatiquement quand l'utilisateur sélectionne une date dans le calendrier.

CheckOnExit : Détermine Si l'utilisateur est autorisé à sortir de la zone de saisie lorsque la date qu'il a saisie n'est pas correcte.

BlankIfNull : Détermine si le composant affiche un texte vide lorsque la valeur de la date est 00/00/0000

ValidDates :

La collection `ValidDates` vous permet d'ajouter un ensemble de valeurs qui seront considérées comme valides lorsque la propriété `CheckOnExit` est true. Il est ainsi possible d'autoriser '00/00/0000' ou ' / / '.

ValidDate : A ne pas confondre avec `ValidDates`. Il s'agit d'une propriété booléenne en lecture seule qui permet de savoir si la date saisie dans le composant est une date valide

Exemple :

```
D mydate          s          D
C                  monitor
C                  eval      mydate=sdGetDate (F1: 'DateEdit1')
C                  on-error
C                  eval      Message=Message+'La date est  invalide'+
C                              + x'0d'
```

Chapitre 14. CCalendar

Onglet : Supplément

Le composant CCalendar permet d'afficher un calendrier.

Ce calendrier représente une date. Cette date est entourée en rouge sur le calendrier.

Pour modifier ou interroger la valeur de la date représentée dans le calendrier, utilisez les fonctions `sdSetDate` et `sdgetDate` ou bien utilisez la propriété `DateIso`.

Il vous est possible de modifier l'image de fond du calendrier à l'aide de la propriété `BackgroundImage`.

Si vous souhaitez revenir à l'image par défaut, supprimez l'image que vous avez ajouté.

Pour ne pas utiliser d'image de fond enlever la valeur `ocBitmap` dans la propriété `Options`.

Chapitre 15. CMaskEdit

Onglet : Supplément

CMaskEdit est un composant qui permet d'offrir une zone de saisie aux utilisateurs. Il est possible de limiter la saisie de l'utilisateur à certains caractères, ou à un format.

La saisie de l'utilisateur est contrôlée selon la propriété EditMask.

EditMask

La propriété EditMask est de type string (chaîne de caractères).

Voici sa description :

EditMask est une chaîne composée de trois champs séparés par un point-virgule.

La première partie du masque est le masque lui-même.

La deuxième partie contient le caractère déterminant si les caractères littéraux du masque sont inclus ou non dans les données.

La troisième partie du masque indique le caractère utilisé pour matérialiser les caractères à saisir dans le masque.

Voici les caractères spéciaux utilisés dans le premier champ du masque :

Caractère	Signification dans le masque
!	Si un caractère ! apparaît dans le masque, les caractères facultatifs sont représentés dans le texte comme des espaces de début. Si le caractère ! n'est pas présent, les caractères facultatifs sont représentés dans le texte par des espaces de fin.
>	Si ce caractère apparaît dans le masque, tous les caractères le suivant sont en majuscules jusqu'à la fin du masque ou jusqu'à occurrence du caractère <.
<	Si ce caractère apparaît dans le masque, tous les caractères qui suivent sont en minuscules jusqu'à la fin du masque ou jusqu'à occurrence du caractère >.
<>	Si ces deux caractères apparaissent ensemble dans un masque, aucune vérification majuscules/minuscules n'a lieu et la donnée est formatée selon les majuscules et minuscules saisies par l'utilisateur.
\	Le caractère suivant ce caractère est un caractère littéral. Utilisez ce caractère lorsque vous souhaitez qu'un caractère spécial du masque soit un caractère littéral inclus dans la donnée.
L	Ce caractère requiert un caractère alphabétique à cette position uniquement. Pour la France, il s'agit de A-Z et a-z.
I	Ce caractère n'autorise qu'un caractère alphabétique à cette position, mais ne l'exige pas.
A	Ce caractère exige un caractère alphanumérique à cette position. Pour la France, il s'agit de A-Z, a-z, 0-9.
a	Ce caractère autorise un caractère alphanumérique à cette position, mais ne l'exige pas.
C	Ce caractère exige un caractère arbitraire à cette position.
c	Ce caractère autorise un caractère arbitraire à cette position, mais ne l'exige pas.
0	Ce caractère exige un caractère numérique à cette position.
9	Ce caractère autorise un caractère numérique à cette position, mais ne l'exige pas.
#	Ce caractère autorise un caractère numérique ou le signe plus ou moins à cette position, mais ne l'exige pas.
:	Ce caractère permet de séparer les heures, les minutes et les secondes. Si le caractère de séparation des heures, des minutes et des secondes défini par les paramètres régionaux du Panneau de configuration de votre système est différent, ce caractère est utilisé à la place de :.
/	Ce caractère permet de séparer les mois, les jours et les années dans les dates. Si le caractère de séparation des jours, des mois et des années défini par les paramètres régionaux du Panneau de configuration de votre système est différent, ce caractère est utilisé à la place de /.
;	Ce caractère sépare les trois champs du masque.
_	Le caractère _ insère automatiquement des espaces dans le texte. Si l'utilisateur saisit des espaces dans le champ, le curseur saute les caractères _.

Tout caractère absent de ce tableau peut apparaître dans le masque comme caractère littéral. Les caractères littéraux doivent avoir un correspondant exact dans la saisie des données du contrôle. Ils sont insérés automatiquement et le curseur les saute pendant la saisie. Les caractères spéciaux peuvent aussi apparaître comme caractères littéraux s'ils sont précédés d'une barre oblique inverse (\).

La deuxième partie du masque ne contient qu'un seul caractère indiquant s'il faut inclure les caractères littéraux du masque dans la propriété Text du contrôle. Par exemple, le masque pour un numéro de téléphone avec indicatif régional pourra être la chaîne suivante :

(000)_000-0000;0;*

Un 0 dans la deuxième partie du masque indique que les littéraux ne doivent pas être inclus dans la propriété Text ; tout autre caractère signifie qu'il faut les inclure. Il est possible de changer le caractère indiquant l'inclusion des littéraux dans l'éditeur de la propriété EditMask ou par programme en modifiant la constante typée MaskNoSave.

La troisième partie du masque indique le caractère apparaissant dans le contrôle de saisie pour matérialiser les blancs (les caractères à saisir). Par défaut, c'est le même caractère que celui des espaces littéraux. Ces deux caractères apparaissent donc identiques dans la fenêtre de saisie. Cependant, si un utilisateur entre des données dans un contrôle de saisie masqué, le curseur sélectionne successivement chaque caractère blanc et saute au-dessus des caractères espace.

Editeur spécifique

Le designer permet de modifier la valeur de la propriété EditMask à l'aide d'un éditeur spécifique.

Cliquez sur le bouton ellipsis dans l'inspecteur de propriétés.

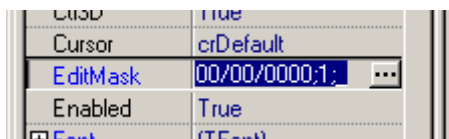
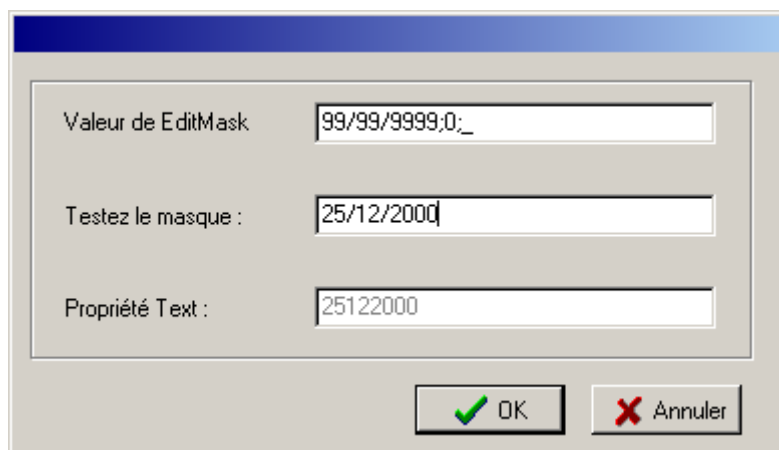


Figure 29

Dans l'éditeur, la première zone permet de saisir la valeur de la propriété EditMask

La seconde zone permet de tester le masque tel que l'utilisateur le verra.

La troisième zone permet de voir le résultat de la propriété text

**Figure 30**

Chapitre 16. CImage

Onglet : Supplément

Le composant CImage permet d'afficher une image.

Design time

Pour modifier l'image contenue dans le CImage, utilisez la propriété Picture.

Un éditeur vous permettra de sélectionner une image sur votre disque.

Les formats bmp, jpg et ico sont supportés.

RunTime

Fichier ifs vers CImage

Pour modifier l'image contenu dans un composant CImage à l'exécution, utilisez la fonction sdSetImg.

cette fonction a besoin de connaître la fiche sur laquelle se trouve le composant CImage, le nom du composant, ma propriété recevant l'image ('picture'), le type d'image, et le chemin de l'image sur l'ifs.

Cette fonction renvoie une valeur

0 : Tout s'est bien passé

-1: Le fichier spécifié n'existe pas ou vous n'avez pas les droits. L'image n'est pas chargée.

-2: Le fichier envoyé ne semble pas être du type indiqué, ou le composant ne peut pas charger ce type d'image.

(Ex : Vous envoyez une image en jpg et vous indiquez qu'il s'agit d'un bmp.)

CImage vers fichier ifs

Pour récupérer le contenu d'une image utilisez la fonction sdGetImg.

Cette fonction a besoin de connaître :

La fiche sur laquelle se trouve le composant.

Le nom du composant.

La propriété contenant l'image ('picture')

L'emplacement sur l'ifs ou vous souhaitez sauvegarder l'image.

La fonction sdGetImg renvoie une valeur :

0: Tout s'est bien passé

-1: L'image dans le composant est vide, et donc le fichier sur l'ifs n'a pas été créé/modifié.

-2: Le nom du fichier ifs est incorrect ou vous n'avez pas les droits suffisants

Vider un Cimage

Pour vider une image, utilisez l'expression suivante :

```
sdSet(Fiche:'Image1':'Picture.Graphic': 'Null')
```

Charger une image depuis le disque de l'utilisateur

Si vous souhaitez que l'utilisateur puisse modifier le contenu du CImage en choisissant une image sur son disque, utilisez les fonctions sdSelectFile et sdLoadFromFile.

Exemple :

```
PSelectImg      B
D               PI
D PForm         5u 0 value
D path          s      1000    varying
C              if      sdSelectFile('*.*jpg |*.jpg':
C                      Path:seOpenPicture)
C              callp    sdLoadFromFile(Fiche:
C                      'Image1.picture':
C                      path)
C              endif
P               E
```

Vous pourrez ensuite récupérer le contenu de l'image par la fonction sdGetImg.

Remarque : La fonction sdSelectFile peut être remplacée par la fonction sdDialog et un composant COpenPictureDialog.

Zoomable

Spécifie si l'utilisateur peut effectuer un zoom sur une partie de l'image.

Pour effectuer le zoom, la propriété stretch doit être à true.

L'utilisateur doit alors sélectionner la partie qu'il veut zoomer avec la souris.

Remarque :

Seules les images de type bitmap, jpeg et gif peuvent être zoomées.

Chapitre 17. CSplitter

Onglet : Supplément

Le composant CSplitter ne nécessite aucun code de votre part.

Ajoutez à une fiche un séparateur entre deux contrôles alignés (propriété align = alLeft ou alRight, etc..) pour permettre aux utilisateurs de redimensionner les contrôles lors de l'exécution. Le séparateur se situe entre un contrôle aligné sur un bord de la fiche et les contrôles remplissant le reste de la zone client. Alignez le séparateur de la même façon que le contrôle ancré sur le bord de la fiche. Quand l'utilisateur déplace le séparateur, il redimensionne le contrôle ancré. Cela change la zone client de la fiche, et les contrôles remplissant le reste de la zone client sont redimensionnés en conséquence.

Utilisez chaque contrôle sur la fiche comme un volet séparé. Après le positionnement de chaque volet, placez un séparateur avec le même alignement pour autoriser le redimensionnement de ce volet. Le dernier volet à placer sur la fiche doit être aligné sur la zone client, afin de se redimensionner automatiquement pour remplir l'espace restant après le redimensionnement de tous les autres volets.

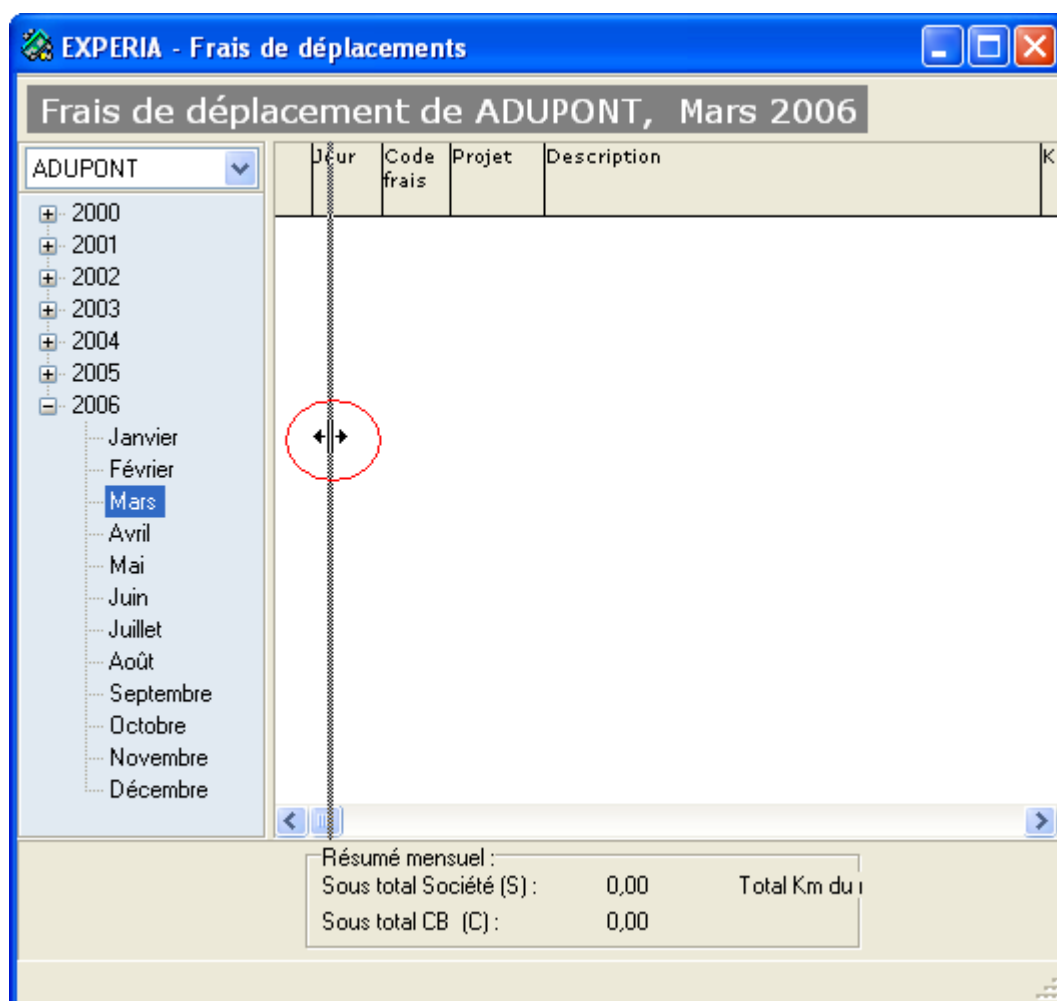


Figure 31

Sens

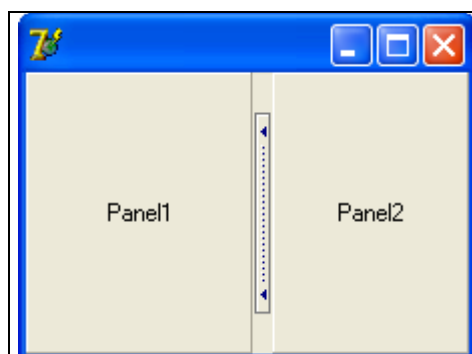


Figure 32

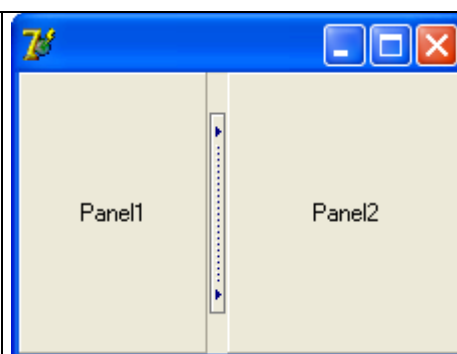


Figure 33

Panel2 en aClient, Panel et splitter en aLeft

Panel1 en aClient, Panel2 et splitter en aRight

MinSize :

Indique la taille minimum, en pixels, des volets de part et d'autre du séparateur.

MinSize permet de définir une taille minimum que le séparateur devra laisser lors du redimensionnement du contrôle adjacent. Si, par Exemple : , la propriété Align vaut alLeft ou alRight, le séparateur de ne peut rétrécir les zones à droite ou à gauche à moins de MinSize pixels. Si la propriété Align vaut alTop ou alBottom, le séparateur de ne peut rétrécir les zones au dessus et en dessous du séparateur à moins de MinSize pixels. La valeur par défaut de MinSize est 30.

Remarque : Affectez toujours à MinSize une valeur inférieure à la moitié de la largeur client de son parent. Quand MinSize est égal à la moitié de la largeur client du parent du séparateur, le séparateur ne peut être déplacé car pour ce faire, il devrait réduire l'un des volets à moins de MinSize pixels.

Chapitre 18. CTreeView

Onglet : Win32

Le composant CTreeView permet d'afficher des données dans un arbre hiérarchique. Chaque noeud du Treeview possède une propriété Text et une propriété key.

Editer en design Time

Dans le designer, pour éditer les noeuds, utilisez l'inspecteur de propriété et utilisez la propriété Items. Il est aussi possible d'effectuer un click droit sur le treeView et de cliquer sur l'élément de menu « editeur ».

Enfin, il est possible de double cliquer sur le composant CTreeView.

Vous arriverez alors sur l'écran suivant :

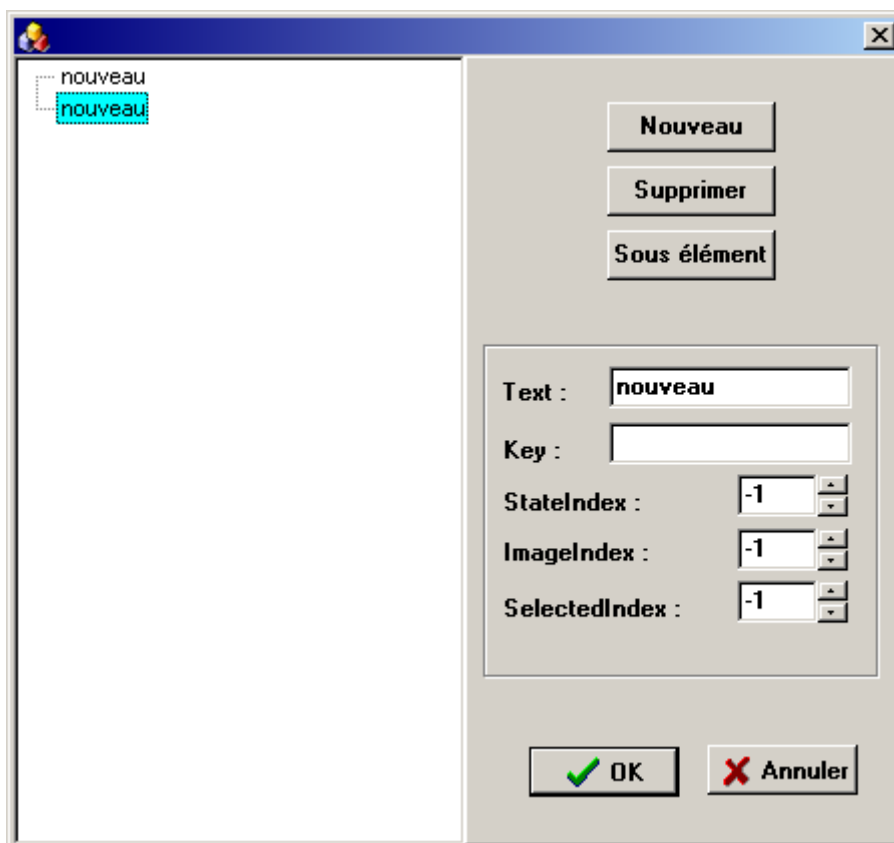


Figure 34

Contrairement aux autres éditeurs spécifiques, cette fenêtre apparaît en modal. Les modifications ne sont répercutés sur le CTreeView que lorsque vous cliquez sur OK.

Editer en run Time

Si les données à afficher dans le TreeView proviennent d'une base de données, utilisez les fonctions `sdCollecClear`, `sdAddNode`, `sdAddNode2` et `sdEndUpd`.

sdCollecClear permet d'effacer tous les éléments.

sdAddNode permet d'ajouter un noeud.

sdAddNode2 permet aussi d'ajouter un noeud. Cette fonction est en général plus rapide que sdAddNode et plus facile d'utilisation. La fonction sdAddNode2 a été créée après la fonction sdAddNode pour améliorer les performances.

La fonction sdAddNode prend en troisième paramètre le numéro du noeud auquel il sera rattaché. (-1 pour ajouter un noeud à la racine)

Remarque : le premier noeud est le noeud d'indice zéro.

```
d sdAddNode      pr
d  pform          5u 0 value
d  Treeview       30   varying value
d  position       10i 0 value
d  Caption        256   varying value
d  Key            256   varying value options(*nopass)
d  ImageIndex     10i 0  value options(*nopass)
d  StateIndex     10i 0  value options(*nopass)
d  SelectedIndex  10i 0  value options(*nopass)
```

```
d sdAddNode2     pr
d  pform          5u 0 value
d  Treeview       30   varying value
d  level          10i 0 value
d  Caption        256   varying value
d  Key            256   varying value options(*nopass)
d  ImageIndex     10i 0  value options(*nopass)
d  StateIndex     10i 0  value options(*nopass)
d  SelectedIndex  10i 0  value options(*nopass)
```

Exemple avec sdAddNode :

```
PLoadTree      b
p              PI
DiPays         s          10  0 inz(0)
DiDep          s          10  0 inz(0)
DCpt           s          10  0 inz(0)
C      KeyDep   klist
C      kfld     IDPAYS
C      kfld     IDDEP
C      callp    sdBeginUpd(F1:'treeView1':'items')
C      callp    sdCollecClear(F1:'treeView1':'items')
C      *loval   setll    pays
```


C		read	pays	55
C		dow	*in55 = *off	
C		callp	sdAddNode(F1:'treeview1':	
C			-1:%trim(NOMPAYS) :'' :0)	
C		eval	iPays = cpt	
C		eval	cpt = cpt + 1	
C	IDPAYS	setll	deps	
C	IDPAYS	reade	deps	56
C		dow	*in56 = *off	
C		callp	sdAddNode(F1:'treeview1':	
C			iPays:%trim(NOMDEP))	
C		eval	iDep = cpt	
C		eval	cpt = cpt + 1	
C	keydep	setll	villes	
C	keydep	reade	villes	57
C		dow	*in57 = *off	
C		callp	sdAddNode(F1:'treeview1':	
C			iDep:%trim(NOMVILL))	
C		eval	cpt = cpt + 1	
C	keydep	reade	villes	57
C		enddo		
C	idpays	reade	deps	56
C		enddo		
C		read	pays	55
C		enddo		
C		callp	sdEndUpd(F1:'treeView1':'items')	
C				
P		E		

Exemple avec sdAddNode 2

pLoadTree	b		
D	PI		
C	KeyDep	klist	
C		kfld	IDPAYS
C		kfld	IDDEP
	*facultatif		
C		callp	sdBeginUpd(F1:'treeView1':'items')
	*Reset de tous les noeuds		
C		callp	sdCollecClear(F1:'treeView1':'items')
C	*loval	setll	pays
C		read	pays
			55
C		dow	*in55 = *off
C		callp	sdAddNode2(F1:'treeview1':
C			0:%trim(NOMPAYS))
C	IDPAYS	setll	deps
C	IDPAYS	reade	deps
			56

```

C      dow      *in56 = *off
C      callp    sdAddNode2 (F1: 'treeview1':
C              1:%trim(NOMDEP))
C      keydep    setll    villes
C      keydep    reade    villes      57
C      dow      *in57 = *off
C      callp    sdAddNode2 (F1: 'treeview1':
C              2:%trim(NOMVILL))
C      keydep    reade    villes      57
C      enddo
C      idpays    reade    deps      56
C      enddo
C      read      pays      55
C      enddo
C      callp    sdEndUpd (F1: 'treeview1': 'items')
pLoadTree      e

```

Index

Quelque soit leur niveau dans la hierarchie, les noeuds ont un index comme le montre la Figure 35

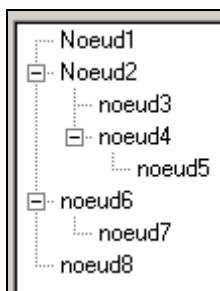


Figure 35

Pour modifier les propriétés d'un noeud en run time, il suffit d'utiliser les fonctions classiques de type sdSet et sdGet.

Pour modifier la propriété Key du noeud numéro 5 par exemple, il faut écrire :

```
sdsetString(f1 : 'treeview1' : 'items(5).key' : 'valeur de la clef')
```

Remarque :

Si une propriété d'un noeud doit être modifiée et que cette propriété ne fait pas partie des paramètres, vous pouvez utiliser la propriété current du composant Treeview. Elle pointe vers le dernier noeud créé.

```
sdsetString(f1 : 'treeview1' : 'current.Key' : 'valeur de la clef')
```

Interroger les éléments sélectionnés.

Le composant CTreeView possède les propriétés FirstSelected et NextSelected qui permettent de connaître la liste des noeuds sélectionnés par l'utilisateur.

FirstSelected et NextSelected renvoient -1 s'il n'y a plus de noeuds sélectionnés.

Il est ensuite possible d'interroger la valeur de la propriété key pour ces éléments.

L'instruction du paragraphe précédent est donc modifiée puisque le numéro du noeud est dans une variable que nous appellerons i.

L'instruction devient :

```
sdsetString(f1 : 'treeview1' : sditem(i) + '.key' : 'valeur de la clef')
```

Remarque 1:

La fonction sdItem() ne fait rien d'autre que de renvoyer une chaîne de caractère.

sditem(0) renvoie 'items(0)'

Remarque 2:

(Voir les propriétés MultiSelect et MultiSelectStyle pour plus de détails).

Exemple :

C	eval	i=sdgetInt(F1:'TreeView1': 'FirstSelected')
C	dow	i <> -1
C	eval	Code= sdGet(F1:'TreeView1':sdItem(i) + '.Key')

...Traitement

C	eval	i=sdgetInt(F1:'TreeView1': 'NextSelected')
C	enddo	

Cases à cocher

Il est possible d'avoir des cases à cocher pour chaque élément du treeview.

Pour cela, modifiez la propriété CheckBoxes du composant.

L'état d'un noeud dépend de la valeur de sa propriété StateIndex.

1 : non coché

2 : coché

3 : coché grisé

Lorsque l'utilisateur coche ou décoche une case, les noeuds parents et enfants sont automatiquement modifiés. Par exemple, si vous cochez un noeud, tous les noeuds enfants seront cochés et le noeud parent se coché si tous les noeuds frères sont cochés et grisés si un des noeuds frère n'est pas coché.

Attention : Cette vérification n'est pas active lorsque vous chargez un treeview par code.

Il est possible d'utiliser les constantes : nsUnchecked, nsChecked et nsGrayed.

Copie en local

Le composant CTreeView autorise l'utilisation des fonctions de type sdGetList.

La fonction sdGetList remonte en local toutes les informations du composant CTreeView. Ces fonctions permettent une optimisation dans le cas où il faut relire tous les noeuds.

Exemple de drag and Drop

Description :

Autorisation ou non d'un drag and drop dans un composant Treeview.

Code

```
procedure TreeView1_OnDragOver (Sender: TObject; Source: TObject; X:
Integer; Y: Integer; State: TDragState; var Accept: Boolean);
var
  myNode:Tnode;
begin
  myNode := FChart.treeview1.getnodeat(X,Y);
  Accept := (myNode <> nil) and (myNode.level = 0) and
    (myNode.parent <> MyNode.Parent);
end;
```

```
pTreeView1_OnDragDrop...
p          B
d          PI
d PevtInf          *   const options(*nopass)
d parm          ds          based(pevtinf)
d Win          5u 0
d evt          48a
d name          100a   varying
d X          10i 0
d Y          10i 0
DNode1          s          10i 0
DNode2          s          10i 0
C          eval          node1=sdgetInt(F1:'TreeView1':
C          'FirstSelected')
C          eval          node2=sdNodeAt(F1:'treeView1':X:Y)
C          if          Node1 = -1 or Node1=Node2
C          return
```

```
C      endif
C      if      sdMsgDlg(boxConfirm:btnYes + btnNo:
C              'Déplacer ce noeud?')=btnYes
C      callp   sdMoveNode(F1:'TreeView1':Nodel:Node
C              'naAddChild')
C      endif
P      E
```

Chapitre 19. CTimer

Onglet : Système

Il n'est pas possible de décider sur le serveur le moment où l'on veut exécuter du code. Le code sur le serveur n'est exécuté que sur une sollicitation du client (c'est à dire lors d'un évènement) Pour réaliser une application qui effectue une tâche cyclique (surveillance de l'arrivée de messages par exemple), vous devrez utiliser le composant CTimer. Ce composant est invisible à l'écran, et il déclenche un évènement cyclique.

Chapitre 20. CTrayIcon

Onglet : Système

Le composant CTrayIcon permet de cacher l'icône d'une application dans la barre des tâches et de la rendre visible dans la zone de notification.

L'évènement onclick est déclenché lorsque l'utilisateur clique sur l'icône qui est dans la zone de notification. Cela permet d'appeler la fonction sdTrayShowMainForm qui fait réapparaître l'icône de l'application dans la barre des tâches et fait apparaître la fenêtre principale.

Pour refaire passer l'application dans la zone de notification, utilisez la fonction sdTrayHideMainForm.

Si vous affectez un popupmenu au composant CTrayIcon, celui sera affiché lorsque l'utilisateur effectuera un click droit sur l'icône dans la zone de notification.

Pour fermer l'application, effectuez un appel à sdClose sur la fenêtre principale comme dans toute application silverdev.

Si vous souhaitez que l'application passe dans la zone de notification lorsque l'utilisateur clique sur la croix, vous devrez écrire un gestionnaire d'evt pour l'evt onclose. Vous devrez aussi ajouter une possibilité à l'utilisateur de fermer l'application par un popupmenu par exemple.

```
pTiQuit_OnClick    B
d                  PI
d PevtInf           *    const options(*nopass)
C                  Eval    CanQuit = *On
C                  callp    sdClose(F1)
p                  E
```

```
pOnClose           B
d                  PI
d PevtInf           *    const options(*nopass)
DParams            ds    based(PevtInf)
D Win              5u 0
D Event            48
D Action           10i 0 overlay(ActionA)
* Le paramètre Action permet de modifier l'action à réaliser :
* 0 Ne rien faire
* 1 Cacher la fenêtre
* 2 Libérer la fenêtre (Mettre variable à 0)
* 3 Minimiser la fenêtre
C                  If    CanQuit
```

```
C      eval      Action=1
C      else
C      eval      Action=0
C      callp     sdTrayHideMainForm(F1: 'TrayIcon1')
C      endif
P      E
```


Chapitre 21. CFontDialog, CColorDialog, COpenDialog, CSaveDialog, COpenPictureDialog, CSavePictureDialog

Ces composants s'utilisent de la même façon.

Utilisez la fonction sdDialog pour afficher la boîte de dialogue associée à ces composants.

Exemples :

```
DName          s          200    varying

C              if          sdDialog(f1:'fontdialog1')
C              eval          name=sdget(f1:'fontdialog1':'font.name')
C              endif
```

```
DCouleur       s          10i 0

C              if          sdDialog(f1:'ColorDialog1')
C              eval          Couleur= sdGetInt(f1:'ColorDialog1':'Couleur')
C              endif
```

```
DFromPage      s          10i 0
DToPage        s          10i 0
Dtype          s          1    inz('1')
DprintRange    s          10i 0

C              if          sdDialog(F1:'printDialog1')
C              eval          printRange=sdGetInt(F1:'PrintDialog1':'PrintRange')
C              if          PrintRange=2
C              eval          FromPage = sdGetInt(F1:'PrintDialog1':'FromPage')
C              eval          ToPage = sdGetInt(F1:'PrintDialog1':'ToPage')
C              endif
C              endif
```

Chapitre 22. CNavBar

Onglet : Standard

Le composant CNavBar permet d'afficher des groupes de liens. Les groupes peuvent être développés ou repliés.

Conception dans le designer

Pour gérer les groupes et les liens dans le composant CNavBar, double cliquez sur le composant, un éditeur spécifique s'ouvre.

Créez un groupe à l'aide du bouton + dans la partie "Groups"

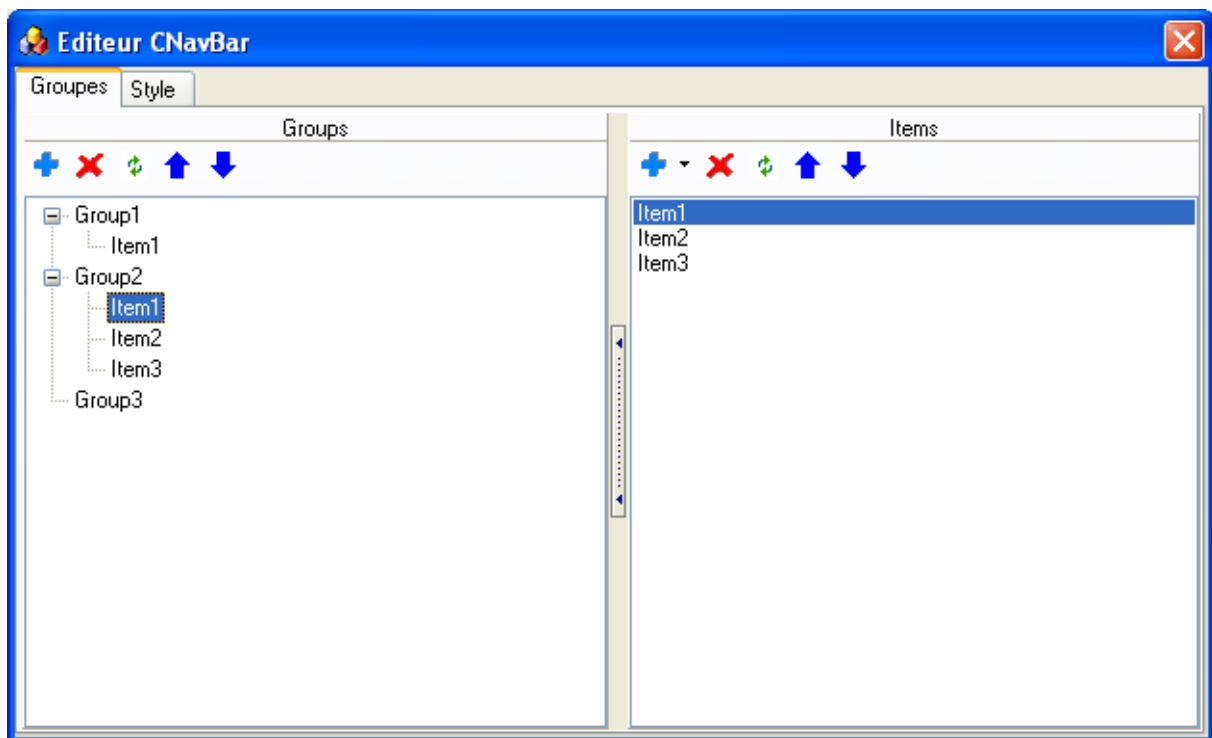


Figure 36

Créez un élément à l'aide du bouton + dans la partie "Items"

Faites glissez un élément de la partie droite dans un groupe de la partie gauche pour associer l'élément au groupe.

Un élément peut être inséré plusieurs fois dans un groupe et peut être inséré dans plusieurs groupes différents.

Pour modifier les propriétés d'un élément ou d'un groupe, sélectionnez l'élément ou le groupe dans l'éditeur de CNavBar, et modifiez les propriétés dans l'inspecteur de propriétés.

Ajout d'un control dans un groupe.

Pour insérer un contrôle dans un groupe du composant CNavBar, modifiez la propriété OptionsGroupControl.UseControl du groupe à true.

Un composant TdxNavBarGroupControl apparaît.

Utilisez ensuite l'arborescence de composants disponible depuis le menu "Outils/Arborescence de composants" pour faire glisser n'importe quel composant visuel dans le groupe.

OnLinkClick

Utilisez l'événement OnLinkClick du composant CNavBar pour effectuer un traitement spécifique à un élément.

L'événement possède un paramètre qui indique le nom de l'élément sur lequel a eu lieu le click.

Remarque : Il est aussi possible d'affecter une action à l'élément.

Création dynamique de groupes.

Exemple de création de groupes et d'éléments :

c	callp	sdNBAddGroup (F1: 'NavBar1': 'group1')
c	callp	sdNBAddItem (F1: 'NavBar1': 'item1')
c	callp	sdNBAddItem (F1: 'NavBar1': 'item2')
c	callp	sdNBAddSeparator (F1: 'NavBar1': 'sep1')
c	callp	sdNBAddLink (F1: 'Group1': 'Item1')
c	callp	sdNBAddLink (F1: 'group1': 'sep1')
c	callp	sdNBAddLink (F1: 'group1': 'item2')

Chapitre 23. CScheduler

Onglet : Scheduler

Le composant CScheduler doit être utilisé conjointement avec un composant CStorage. Le composant CScheduler affiche les événements stockés dans le composant CStorage. Pour lier un composant CScheduler et un composant CStorage, utilisez la propriété Storage du composant CScheduler.

Ajouter les événements

Pour supprimer tous les événements, utilisez la fonction sdClear.

Pour ajouter un événement dans le composant CStorage, utilisez la fonction sdNewEvent.

Les fonctions sdBeginUpdate et sdEndUpdate doivent être appelées respectivement avant et après l'ajout d'événements.

```
c      callp      sdBeginUpd(F1: 'Storage1')
c      callp      sdClear(F1: 'Storage1')
c      callp      sdNewEvent(F1: 'Storage1':A_DateStr:
c                  A_TimeStr:A_DateEnd:A_TimeEnd)
c      callp      sdEndUpd(F1: 'Storage1')
```

Propriétés d'un événement

Après avoir ajouté un événement, il est possible de faire référence à cet événement avec la propriété current du composant CStorage.

```
C      callp      sdSetString(F1: 'Storage1':
C                  'Current.Caption': 'exemple')
```

Les propriétés intéressantes pour un événement sont :

DateStart	integer	Date de début de l'évènement
DateEnd	integer	Date de fin de l'évènement
TimeStart	integer	Heure de début de l'évènement
TimeEnd	integer	Heure de fin de l'évènement
LabelColor	TColor	Couleur de fond de l'évènement
Caption	String	Titre de l'évènement

Message	String	Texte de l'événement
Location	String	emplacement
Field	Variable	Propriété indexée. Le nombre dépend du nombre de champs dans customFields de CStorage. Le premier élément a l'indice 0.
State	Integer	De 0 à 3. Indique la disponibilité. Dans la vue jour, les bords de l'évènement changent d'apparence en fonction de la propriété State.
ResourceId	Variable	Ressource de l'événement.

Valeurs de State :

Valeur	Apparence	Signification
0	Blanc	Libre
1	Bleu et blanc	Temporaire
2	Bleu	Occupé
3	Violet	Déplacement

Valeurs supplémentaires pour un événement

En plus des propriétés fixes , il est possible d'ajouter des propriétés pour chaque événement.

Pour cela , utilisez la propriété customFields du composant CStorage.

Ces propriétés pourront être accéder à l'aide de la propriété indexée Field.

C	callp	sdSetString(F1: 'Storage1':
C		'Current.Field(0) ': 'exemple')

La première propriété dans customFields a l'index 17, la suivante l'index 18, etc..

Utilisez la première propriété de customFields pour insérer les valeurs identifiant l'évènement, car la valeur de cette propriété est transmise dans les différents évènements du scheduler

DateNavigator

Le composant affiche un calendrier de navigation afin que l'utilisateur puisse se déplacer dans son agenda.

Ce calendrier peut afficher les jours qui possèdent des évènements selon la propriété DateNavigator.ShowDatesContainingEventsInBold.

Dans le cas ou cette propriété est true, il faut bien sur ajouter les éléments avec la fonction sdNewEvent sur toute la période du calendrier.

Sélectionner une période par programme

D	sdSelectDays	pr	
D	F1	5u 0	const
d	Component	30	varying value
D	DateStr	8 0	value
D	DateEnd	8 0	value
D	ViewDay	N	value

Pour connaître les dates de début et de fin du calendrier, interrogez les variables NavigatorDateStart et NavigatorDateEnd du composant CScheduler.

Différentes vues

Le composant CScheduler possède différentes vues. Ces vues sont automatiquement sélectionnées selon la période qui est choisi dans le DateTimeNavigator.

Vous pouvez désactiver des vues, par exemple pour que la vue semaine ne s'affiche pas, modifiez la propriété viewWeek.CanShow à false.

Ressources

Pour afficher plusieurs ressources en même temps, ajoutez des éléments dans la collection CStorage.Resources.Items

Utilisez la propriété ResourceId d'un événement pour affecter l'événement à cette ressource.

Exemple

C	callp	sdCollecAdd(F1: 'Storage1':
C		'Resources.Items')
C	callp	sdSetString(F1: 'Storage1': 'resources.' +
C		'Items.Items(0).ResourceId: 'Res1')
C	callp	sdSetString(F1: 'Storage1':
C		'Current.ResourceId: 'Res1')

Fiches et popupMenu internes

Le composant CScheduler possède des fiches et des popupmenu internes.

Dans la plupart des cas, vous souhaitez ne pas utiliser ces éléments pour utiliser vos fiches et popupmenus personnalisés. Dans ce cas, pensez à modifier les propriétés suivantes :

```

EventOperations.DialogEditing
EventPopupMenu.UseBuiltInPopupMenu
ContextPopupMenu.UseBuiltInPopupMenu
ViewDay.TimerPopupMenu.items

```

Drag and drop d'un événement

Pour pouvoir déplacer un événement, modifiez la propriété `EventOperations.moving` à `true`.

Interceptez l'événement `OnAfterDragEvent` afin de refléter les modifications dans la base.

Il est possible d'annuler le déplacement avec le paramètre `Accept`.

Les nouvelles informations de l'événement sont dans les paramètres `Datas`, `StrDate`, `StrTime`, `EndDate`, `EndTime`.

`Datas` contient les valeurs insérées dans le premier élément des `customFields`.

```

~/EVENT Scheduler1_OnAfterDragEvent
, *-----*
, * Description : *
, *-----*
D Parameters      ds              based(pevtfinf)
D Win              5u 0
D Evt              48a
  * Modifiable
D Accept           N
  * Non modifiables
D Datas            100      varying
D StrDate          10u 0
D StrTime          10u 0
D EndDate          10u 0
D EndTime          10u 0
  *

```

Exemple :

```

c              eval      msg = 'Confirmez vous le déplacement ?'+
c              X'0d25'+ 'Nouveau créneaux :'+X'0d25'+
c              'De '+DspDate(StrDate)+' ' +
c              DspHour(StrTime) + ' à ' +
c              DspDate(EndDate)+' ' +DspHour(EndTime)
C              eval      Accept = sdMsgDlg(boxConfirm:
c              btnOK + btnCnl:msg) = btnok

```

```

c      if      not Accept
c      return
c      endif

c      eval    dsInfos = Datas
C      move    dsNoevt      A_noevt
C      A_noevt chain    sddmevts
c      if      %found(sddmevts)
c      eval    A_DateStr= StrDate
c      eval    A_DateEnd = EndDate
c      eval    A_TimeStr = StrTime
c      eval    A_TimeEnd = EndTime
c      update  sddmevtsF
c      endif

```

Il est aussi possible d'empêcher le déplacement dans l'événement onBeforeDragEvent grâce au paramètre Allow.

```

D Parameters      ds                      based(pevtinf)
D Win              5u 0
D Evt              48a
,* Modifiable
D Allow            N
,* Non modifiables
D Datas            100      varying
,*

```

Modifier la durée d'un événement

Pour pouvoir modifier la durée d'un événement directement dans l'agenda, modifiez la propriété EventOperation.sizing à true.

Interceptez l'événement OnAfterSizingEvent afin de refléter les modifications dans la base.

Il est possible d'annuler la modification avec le paramètre Accept.

Les nouvelles informations de l'événement sont dans les paramètres Datas, StrDate, StrTime, EndDate, EndTime.

Datas contient les valeurs insérées dans le premier élément des customFields

```

D Parameters      ds                      based(pevtinf)
D Win              5u 0
D Evt              48a
,* Modifiable
D Accept            N
,* Non modifiables
D Datas            100      varying
D StrDate          10u 0

```



```
D StrTime          10u 0
D EndDate          10u 0
D EndTime          10u 0
*
```

Il est aussi possible d'empêcher le déplacement dans l'événement `onBeforeSizingEvent`

```
D Parameters      ds          based(pevtinf)
D Win             5u 0
D Evt             48a
,* Modifiable
D Allow           N
,* Non modifiables
D Datas           100         varying
```

Supprimer un événement

Pour pouvoir supprimer un événement directement dans le calendrier, modifiez la propriété `EventOperation.deleting` à `true`.

Interceptez l'événement `OnBeforeDeleting`

`Datas` contient les valeurs insérées dans le premier élément des `customFields`

Le paramètre `Allow` permet d'annuler l'opération.

```
D Parameters      ds          based(pevtinf)
D Win             5u 0
D Evt             48a
,* Modifiables
D Allow           N
,* Non modifiables
D Datas           100         varying
*
C                  eval      Allow = sdMsgDlg(boxConfirm:
c                  btnOK + btnCnl:
c                  'Supprimer l''évènement ?') = btnok
```

Double clique sur un événement

Utilisez le paramètre `Datas` pour retrouver l'enregistrement correspondant dans la base de données.

```
D Parameters      ds          based(pevtinf)
```

D Win	5u 0	
D Evt	48a	
D Datas	100	varying

Double clique sur l'agenda

Si vous souhaitez afficher une fenêtre de création d'un événement à la date et à l'heure ou a eu lieu le double click, utilisez les paramètres de l'événement :

D Parameters	ds	based(pevtinf)
D Win		5u 0
D Evt		48a
D Date		10u 0
D Time		10u 0
D Index		10i 0

Détecter le changement de période

L'événement onPeriodChanged permet de détecter que la période affichée par les calendriers a changé. Si la propriété DateNavigator.ShowDatesContainingEventsInBold est à true, il faut alors relire la base.

L'événement onSelectionPeriodChanged permet de détecter que la période sélectionnée dans le calendrier a changé. Si la propriété DateNavigator.ShowDatesContainingEventsInBold est à false il est possible de relire la base dans cet événement pour rafraichir l'agenda.

Les deux événements ont les même paramètres :

D Parameters	ds	based(pevtinf)
D Win		5u 0
D Evt		48a
D Deb		10u 0
D end		10u 0

Images dans un événement

Pour ajouter une image dans un événement, affectez la propriété eventImages. Ajoutez des informations à l'aide de la propriété customFields.

Utilisez l'événement onInitEventImages en local comme dans l'exemple ci-dessous :

```
var
temp:variant;
begin
temp:=AEvent.Field[1];
if not VarIsNull(temp) and (temp='Y') then
begin
AImages.add(0);
end;
temp:=AEvent.Field[2];
if not VarIsNull(temp) and (temp='Y') then
begin
AImages.add(1);
end;
end;
```

Couleur d'un événement

Vous pouvez modifier la couleur d'un événement après l'avoir créé comme dans l'exemple ci dessous :

C	callp	sdSetInt(F1:'Storage1':
C		'Current.labelColor':X'FF00FF')

Programme exemple sddmview

Dans silverdemo , le programme SDDMVIEW fournit un exemple de l'utilisation du composant CSCHEDULER.

Chapitre 24. CLinkLabel

CLinkLabel est un composant qui permet de placer un lien vers une page internet ou un envoi de mail.

La propriété Link contient toutes les caractéristiques spécifiques au CLinkLabel par rapport au CLabel.

Link.Color :

Représente la couleur que prend le fond lors du passage de la souris.

Link.Font :

Représente les caractéristiques de la police lors du passage de la souris

Link.Url :

Représente le liens vers lequel l'utilisateur sera dirigé lors d'un click

Remarque :

*Pour envoyer vers une page internet, utiliser une url du type : `http :
//www.experia.com`*

*Pour envoyer vers un envoi de mail, utiliser une url du type : `mailto :
infos@experia.com`*

Remarque :

Le composant ouvre le document renseigné dans la propriété link seulement si l'événement onclick n'est pas utilisé. Si l'événement onclick est renseigné aucune action n'est réalisée automatiquement.

Chapitre 25. CCmdDialog

Description

Le composant CCmdDialog permet d'afficher une invite de commande.

Le composant génère dynamiquement une fenêtre contenant les différents paramètres de la commande.

Utilisez la fonction sdCmd pour afficher l'invite de commande.

Exemple :

```
D cmd          s          1000    varying
D State        s          10u 0
c              eval       cmd = sdGet(F1:'edit1':'text')
c              callp      sdSetString(F1:'CmdDialog1':'Command':cmd)
c              eval       state = sdCmd(F1:'CmdDialog1')
c              eval       cmd = sdGet(F1:'CmdDialog1':'Command')
```

Si le troisième paramètre de la fonction contient une fonction valide, l'invite de commande est construite à partir de cette expression.

Si par exemple, le troisième paramètre contient l'expression

"CRTBNDRPG PGM(SILVERDEMO/SDTSTCMD)

SRCFILE(SILVERDEMO/QRPGLESRC)", la fenêtre affichée sera la suivante :

***LIBL/CRTBNDRPG**

Prompt | Texte

ACTGRP
ALWNULL
AUT
BNDDIR
CVTOPT
DBGVIEW
DEFINE
DFTACTGRP
ENBPFRCOL
FIXNBR
GENLVL
INCDIR
INDENT
INFOTMF
LANGID
LICOPT
OPTIMIZE
OPTION
OUTPUT
PGM
PGMINFO
PPGENOPT
PPSRCFILE
PPSRCMBR
PPSRCSTMF
PRFDTA
REPLACE
SRCFILE
SRCMBR
SRCSTMF
SRTSEQ
TEXT
TGTRLS
TRUNCNBR
USRPRF

PGM
Program: Name, *CTLSPEC
Library: Name, *CURLIB

SRCFILE
Source file: Name, QRPGLSRC
Library: Name, *LIBL, *CURLIB

SRCMBR
Source member: Name, *PGM

SRCSTMF
Source stream file:

GENLVL
Generation severity level: 0-20

TEXT
Text 'description': Character value...

DFTACTGRP
Default activation group: *YES, *NO

ACTGRP
Activation group: Name, QILE, *NEW, *CALLER

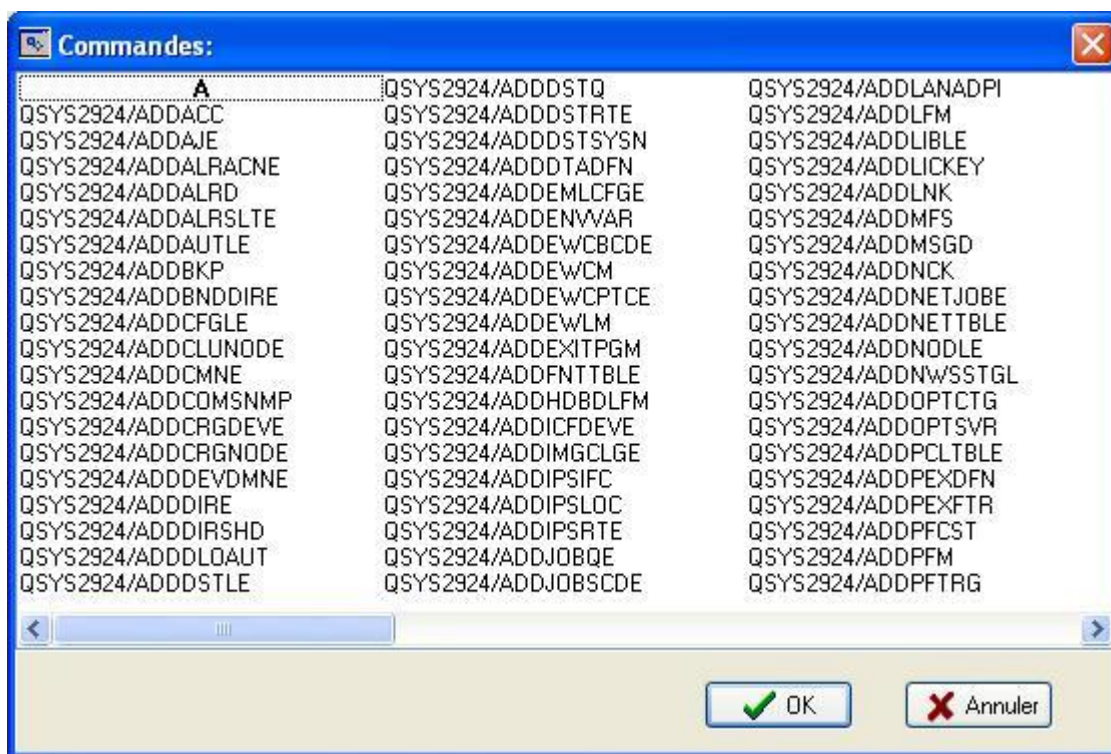
BNDDIR
 Name

Sélection de commande :

Si le troisième paramètre est vide et que la propriété AllowSelection est à true, alors une première fenêtre de choix de commande apparaît :



Les boutons à droite de bibliothèque et Commande permettent de choisir dans une liste.



Si la propriété ExecuteCmd est à true, alors la commande est exécuté lorsque l'utilisateur clique sur ok.

Si la propriété ShowErrorLog est à true, le job log généré par la commande est affichée comme dans l'exemple suivant :

Job log								
Identifiant	Rang	Message	Aide	Fichier de message	Bibliothèque fichier de message	Procédure réception	Date envoi	Heure envoi
RNS9310	1	Compilation failed. Program SDTSTCM not created in library SILVERDEMO.	Cause : Compilation failure normally occurs when the severity of issued messages exceeds the value specified for the GENLVL parameter on the CRTBNDRPG command. See the compiler listing or the job log for error messages. Recovery : Correct the errors or change the value specified for the GENLVL parameter. Compile again.	QRPGLEMSG	QSYS2924	BPCMD	26/05/2010	16:52:39
RNS9306	2	Unable to open member SDTSTCM of file QRPGLSRC in library SILVERDEMO.	Cause : The error is due to one of the following: -- the source member SDTSTCM of file QRPGLSRC in library SILVERDEMO does not exist -- the source file QRPGLSRC in library SILVERDEMO does not exist -- the library SILVERDEMO cannot be found -- the file cannot be opened at this time. See previously listed messages in the job log. Recovery : Correct the error and compile again.	QRPGLEMSG	QSYS2924	BPCMD	26/05/2010	16:52:39
CPF4102	3	File QRPGLSRC in library SILVERDEMO with member SDTSTCM not found.	Cause : The file was not opened. The reason code is 05. The reason codes and their meanings are as follows: 01 - The library does not exist. 02 - The file does not exist. The library does exist. 03 - The file does not exist. The library specified as *LIBL. 04 - The member was saved with storage freed. 05 - The member does not exist. The file does exist. 06 - The file does not have any members. 10 - The member on the remote system is not found. 14 - The member was saved with storage freed on the remote system. 15 - The member on the remote system does not exist. The file does exist. 16 - The file on the remote system does not have any members. Recovery : Do one of the following based on the reason code shown, and try the request again: 01 - Change the library name with an Override Database File (OVRDBF) command. 02 - Change the file name, the library name, or both the file name and library name with an Override Database File (OVRDBF) command. 03 - Add the library to the library list (ADDLIBL command) or change the file name, the library name, or both the file name and library name with an Override Database File (OVRDBF) command. 04 or 14 - Restore the file that contains the member (RSTOBJ command) or delete the file (DLTF command) and recreate the file (CRTLF, CRTPF, or CRTSRCPF command). 05 or 06 - Add a member (ADDLFM or ADDPFM command) or change the file name, the library name, or the member name with an Override Database File (OVRDBF) command. 10, 15 or 16 - Add a member (ADDLFM or ADDPFM command) or change the file name, the library name, or the member name in the distributed data management (DDM) file with a Change DDM File (CHGDDMF) command.	QCPFMSG	QSYS2924	C_Open file	26/05/2010	16:52:39

Si la propriété ShowErrorLog est à false, vous pouvez récupérer vous même le job log généré à l'aide de l'api qmhljobl.

Onglet Texte

La fenêtre affichant l'invite de commande contient deux onglets. Dans le second onglet, il est possible de modifier la commande en mode texte.

Il est possible à tout moment de passer d'un onglet à l'autre.

Cependant, la saisie en mode texte peut entraîner une erreur (commande incorrecte).

Une fenêtre d'erreur est alors affichée. Dans certains cas, l'erreur est trop importante pour permettre de repasser en mode invite.

L'onglet Texte permet de changer totalement le texte de la commande et donc la commande elle même. Si vous souhaitez désactiver la possibilité de modifier le texte dans l'onglet Texte, attribuez la valeur false à la propriété AllowText.

Paramètre de type commande

Certaines commandes ont des paramètres de type commande. Par exemple le paramètre CMD de la commande SBMJOB est lui même une commande.

Dans ce cas, il est possible d'utiliser la touche F4 sur la zone pour obtenir une nouvelle invite de commande.

Propriétés

Nom	Type	Description
AllowSelection	Boolean	Indique si une fenêtre de selection de commande apparait lorsque la propriété Command est à blanc.
AllowText	Boolean	Indique si l'utilisateur peut modifier la commande dans l'onglet Texte.
Command	String	Indique la commande à afficher. Cette propriété est modifiée lorsque l'utilisateur clique sur ok dans l'invite de commande.
ExecuteCmd	Boolean	Indique si la commande est exécutée lorsque l'utilisateur cliquer sur ok. Si ExecuteCmd est à false, il appartient au développeur d'exécuter la commande avec les api qcmandexc ou qcpcmd par exemple.
ShowErrorLog	Boolean	Indique si les messages de log sont affichés en cas d'erreur dans la commande. Cette propriété n'a pas d'effet si ExecuteCmd est à false.
ShowLastLog	Boolean	Indique si un message d'erreur est affiché en cas d'erreur dans la commande. Cette propriété n'a pas d'effet si ExecuteCmd est à false.
AllowSelection	Boolean	Indique si une fenêtre de selection de commande apparaît lorsque la propriété Command est à blanc
State	TStateCmdDialog	Indique le résultat de l'invite de commande.

Valeurs de State :

Valeur	Description
0	Inconnu
1	L'utilisateur a cliqué sur ok et ExecuteCmd est à false
2	L'utilisateur a cliqué sur ok, ExecuteCmd est à true et la commande est exécutées sans erreur.
3	L'utilisateur a cliqué sur Annuler.
4	La commande passée dans la propriété Command est incorrecte.
5	L'utilisateur a cliqué sur ok, ExecuteCmd est à true, et la commande a générée une erreur.

Evènements :

Les évènements onCommand et onErrorCommand vous permettent de créer des fenêtres personnalisées. Il est possible d'utiliser l'api QMHLJOB1 pour retrouver les messages dans le job log. La clef du premier et du dernier messages générés pendant l'exécution de la commande sont donnés en paramètres dans la commande.

CMapPoint

Le composant CMapPoint est un composant permettant de manipuler l'application de cartographie MapPoint. Il n'est pas possible d'utiliser ce composant sur un pc ou MapPoint n'est pas installé.

Graphiques

Chapitre 26. CChart

Onglet : Supplément

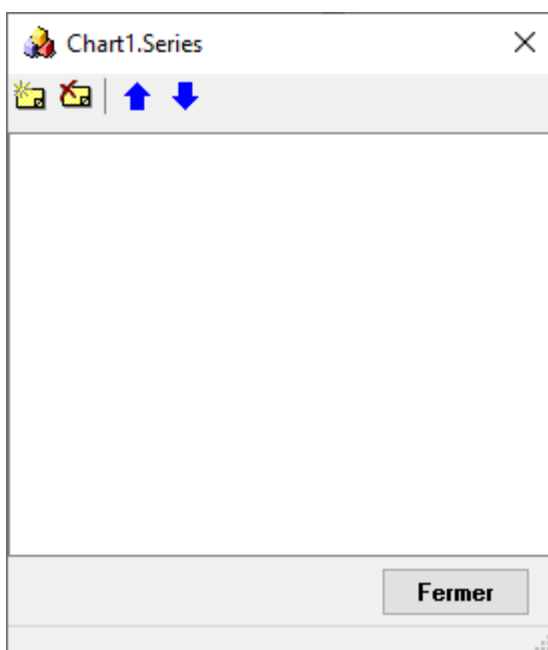
Le composant CChart permet de dessiner des graphes.

Pour ajouter un graphique dans une fiche silverdev, sélectionnez le composant CChart dans l'onglet graphiques de la palette de composants.

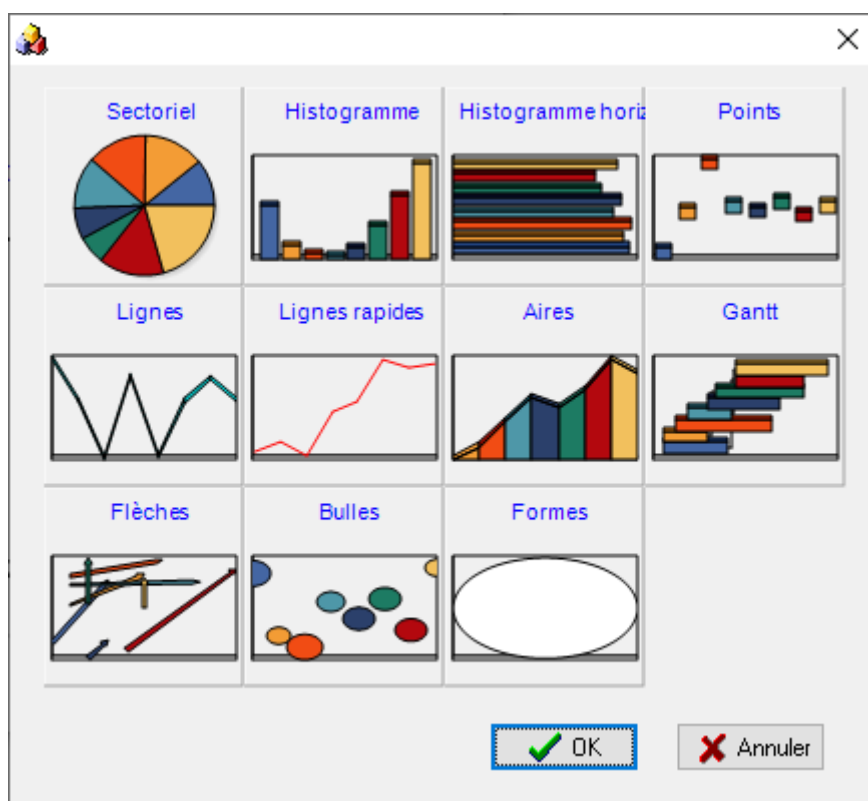
Chapitre 27. TChartSeries

Le composant CChart peut afficher plusieurs graphiques.
Chaque graphique correspond à un composant CSerie.

Les composant de type TChartSeries ne sont pas dans la palette de composants, il faut faire un click droit sur un composant CChart, et choisir « editeur »



La liste des séries contenues dans le cchart est affichée.
Cliquez sur le bouton en haut à gauche pour ajouter une série.



Cliquez sur le type de série désiré.

Les différents types de série sont : TPieSeries, TBarSeries, THorizBarSeries, TPointSeries, TLineSeries, TFastLineSeries, TAeraSeries, TGanttSeries, TChartShape, TArrowSeries, TBubbleSeries.

La propriété Active permet d'activer ou désactiver une série.

Fonctions

Pour supprimer toutes les données dans une série, utilisez la fonction sdSeriesClear, quel que soit le type de serie.

Pour ajouter des données dans une série, utilisez la fonction correspondant selon le type de serie :

TPieSeries	sdAddPie
TBarSeries	sdAddBar
TLineSeries TAreaSeries TFastLineSeries TPointSeries	sdAddXY

TArrowSeries	sdAddArrow
TBubbleSeries	sdAddBubble
TGanttSeries	sdAddGantt

Exemple :

C		callp	sdSeriesClear (F1: 'Series1')	
C	*loval	setll	SDDMTHM	80
C		read	SDDMTHM	80
C	*in80	doweq	*off	
C		Eval	cpt=0	
C	T_IDTHEME	setll	SDDMLIV1	70
C	T_IDTHEME	reade	SDDMLIV1	70
C	*in70	doweq	*off	
C		Eval	cpt=cpt+1	
C	T_IDTHEME	reade	SDDMLIV1	70
C		enddo		
C		callp	sdAddPie (F1: 'Series1':cpt:	
C			T_NOMTHEME)	
C	read	SDDMTHM		80
C	enddo			

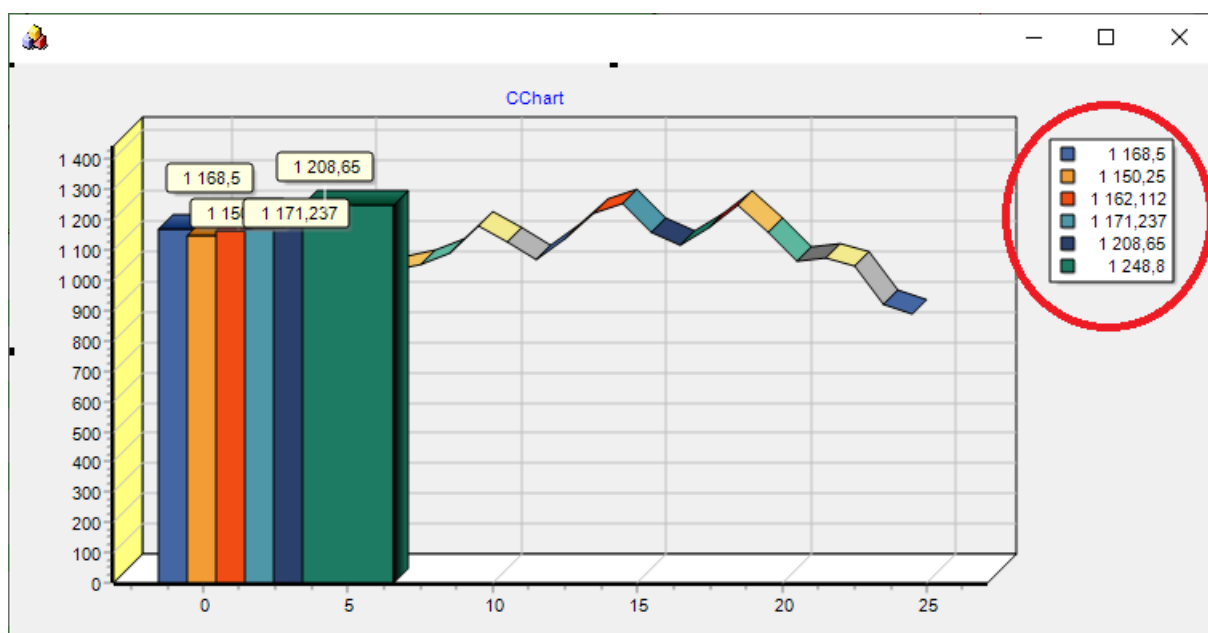
Propriétés

Une des difficultés de l'impression des graphiques est le nombre important de propriétés dans les composants.

Certaines propriétés sont sur le composant CChart, et d'autres sur les composants TChartSeries.

Legendes

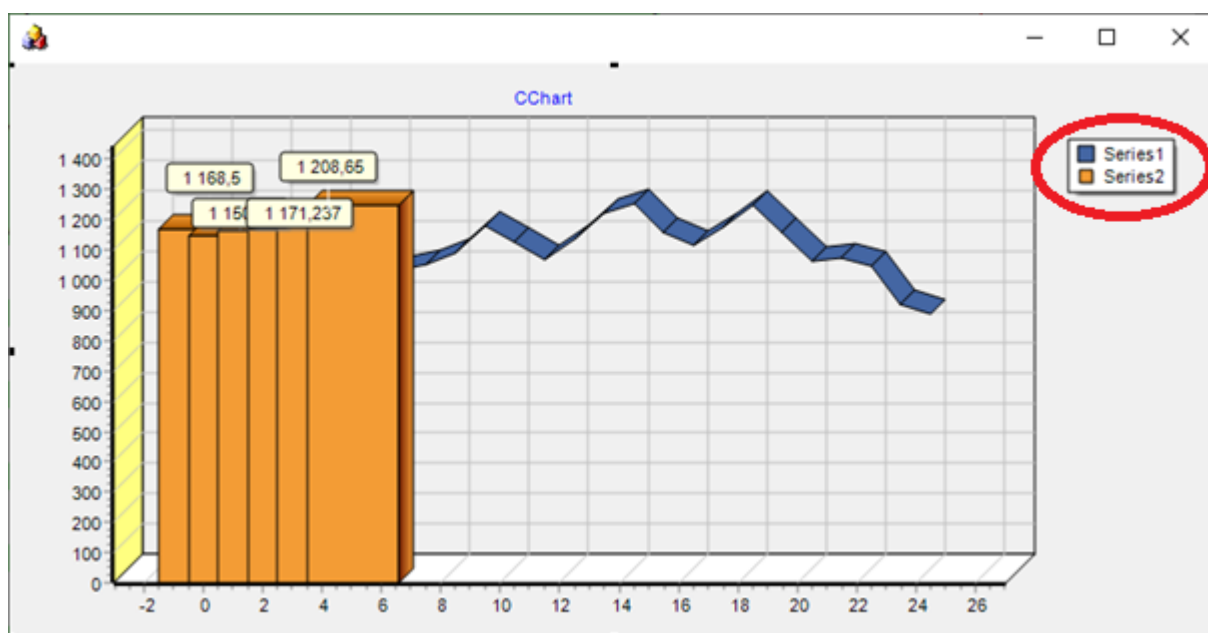
La propriété legend correspond à la partie entourée ci-dessous.



Utilisez la propriété **CChart.Legend** et la propriété **TChartSeries.Legend**

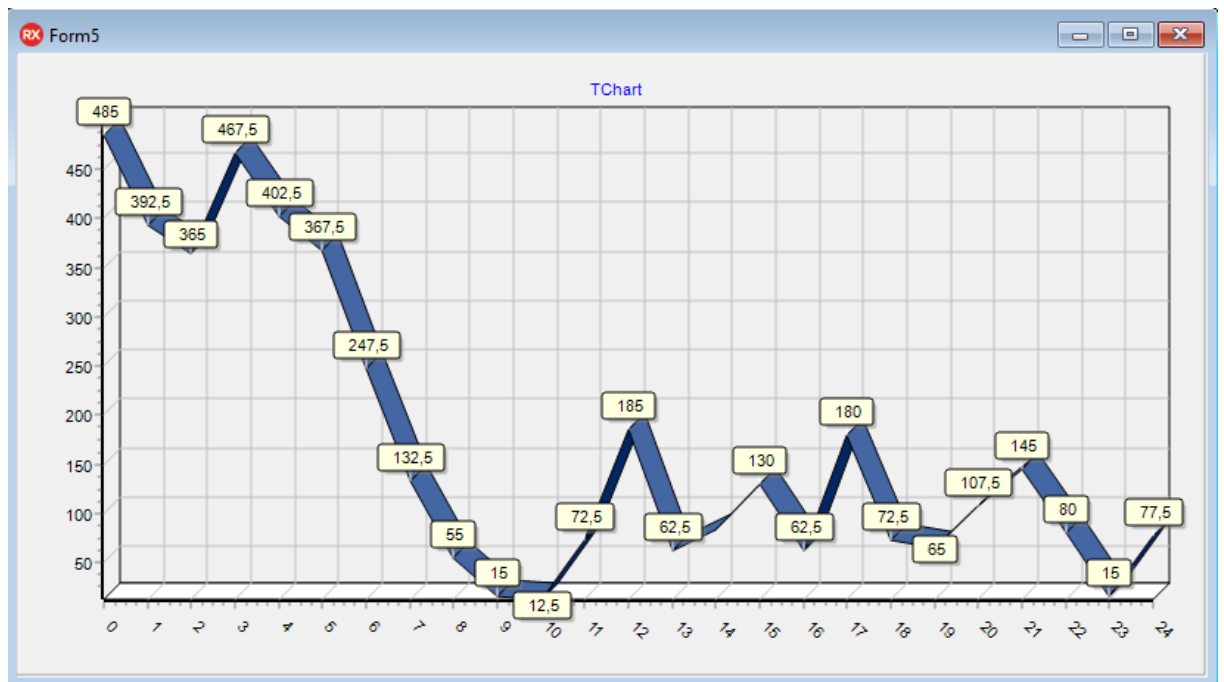
La propriété **CChart.Legend.LegendStyle** permet de choisir d'afficher les valeurs des séries, les noms des series, ou les dernières valeurs de chaque série

Si **CChart.Legend.LegendStyle** est à **IsAuto**, et que plusieurs séries ont la propriété **Legend.visible** à **true**, la légende est ainsi :



Etiquettes

Les étiquettes sont les petits carrés jaunes sur le schema ci dessous :



Les étiquettes sont paramétrables avec la propriété `TChartSeries.Marks`

`TChartSeriesMarks` est une propriété de type `TSeriesMarks`, qui contient notamment les propriétés suivantes :

Arrow :

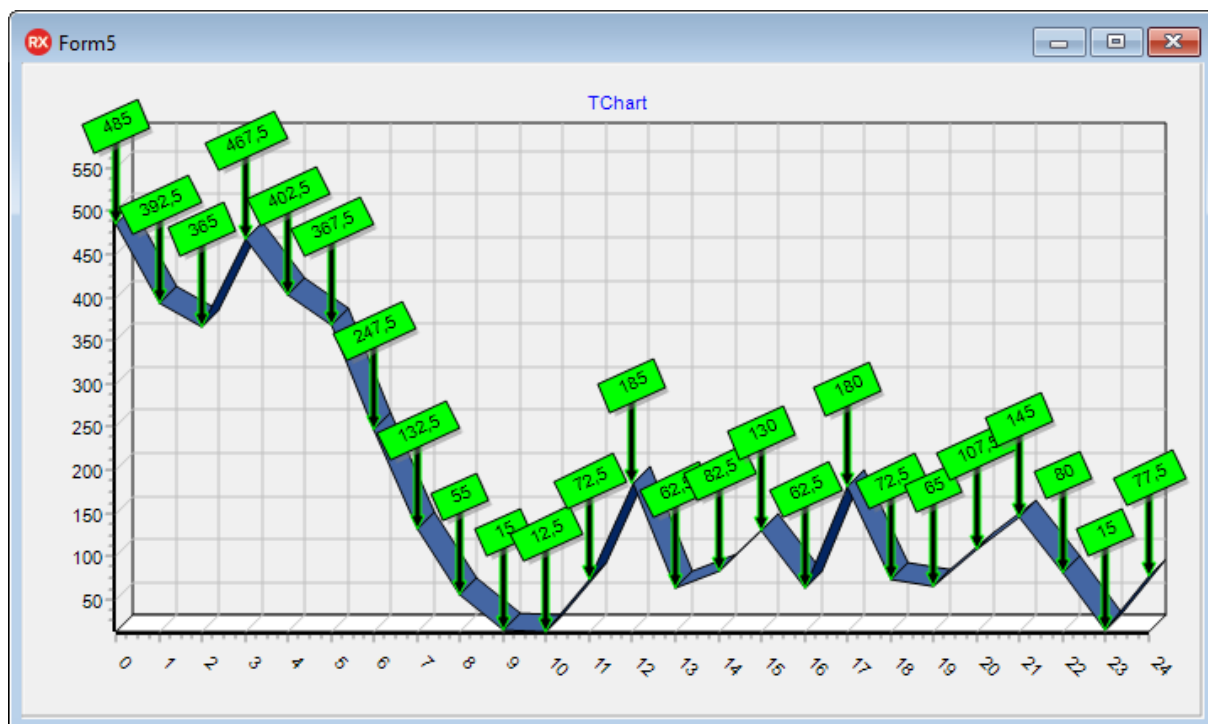
Permet de modifier l'aspect de la fleche qui relie les étiquettes.

ArrowLength :

Longueur de la fleche entre l'étiquette et le point sur le graphique.

Color :

Couleur du fond de l'étiquette



Titres

Les titres se trouvent au niveau du CChart.

Utilisez les propriétés **CChart .Title**, **CChart .SubTitle**, **CChart .Footer** et **CChart .SubFooter**



Chacun de ces titres a les même propriétés.

Pour laisser un espace plus grand entre le titre et le graphe par exemple, ajoutez des lignes vides dans le texte.

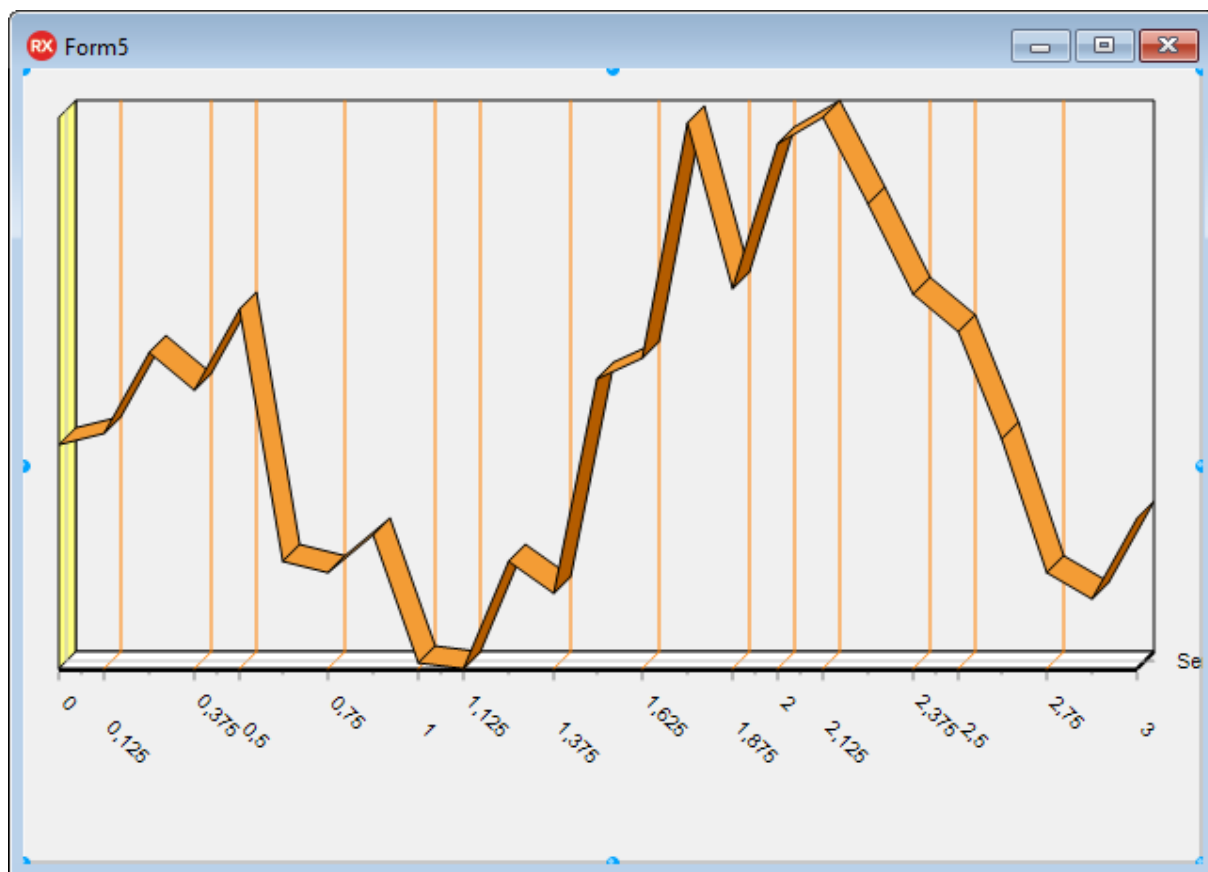
Axes

Pour modifier les axes, utilisez les propriétés **CChart.BottomAxis**, **CChart.LeftAxis**, **CChart.RightAxis** et **CChart.TopAxis**.

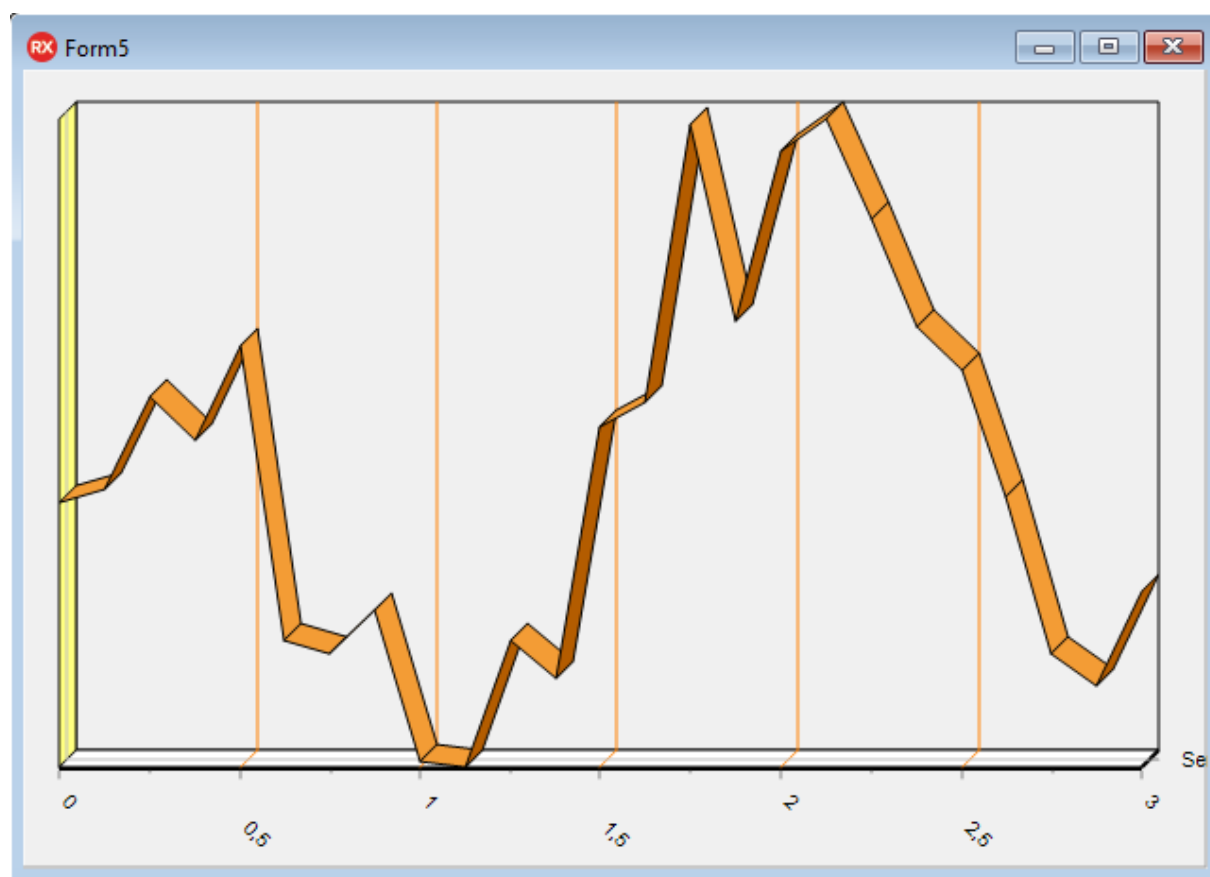
Ces 4 propriétés sont du même type, TChartAxis et ont donc les même sous propriétés.

labelStyle

CChart.xxxxAxis.labelStyle permet de choisir le texte affiché. Par exemple, la valeur `talPointValue` affichera des valeurs uniquement là où il y a des points :

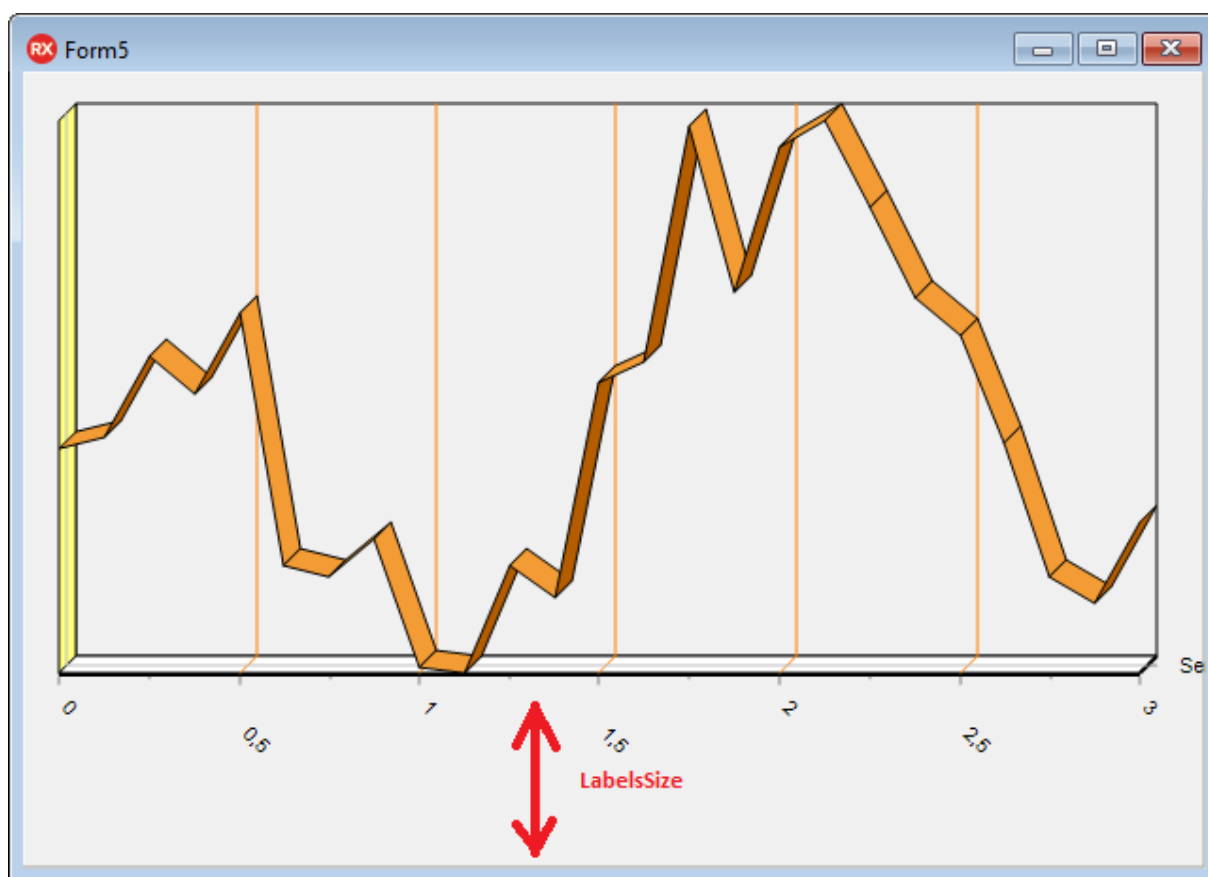


Alors que la valeur `talValue` affichera des valeurs régulièrement :



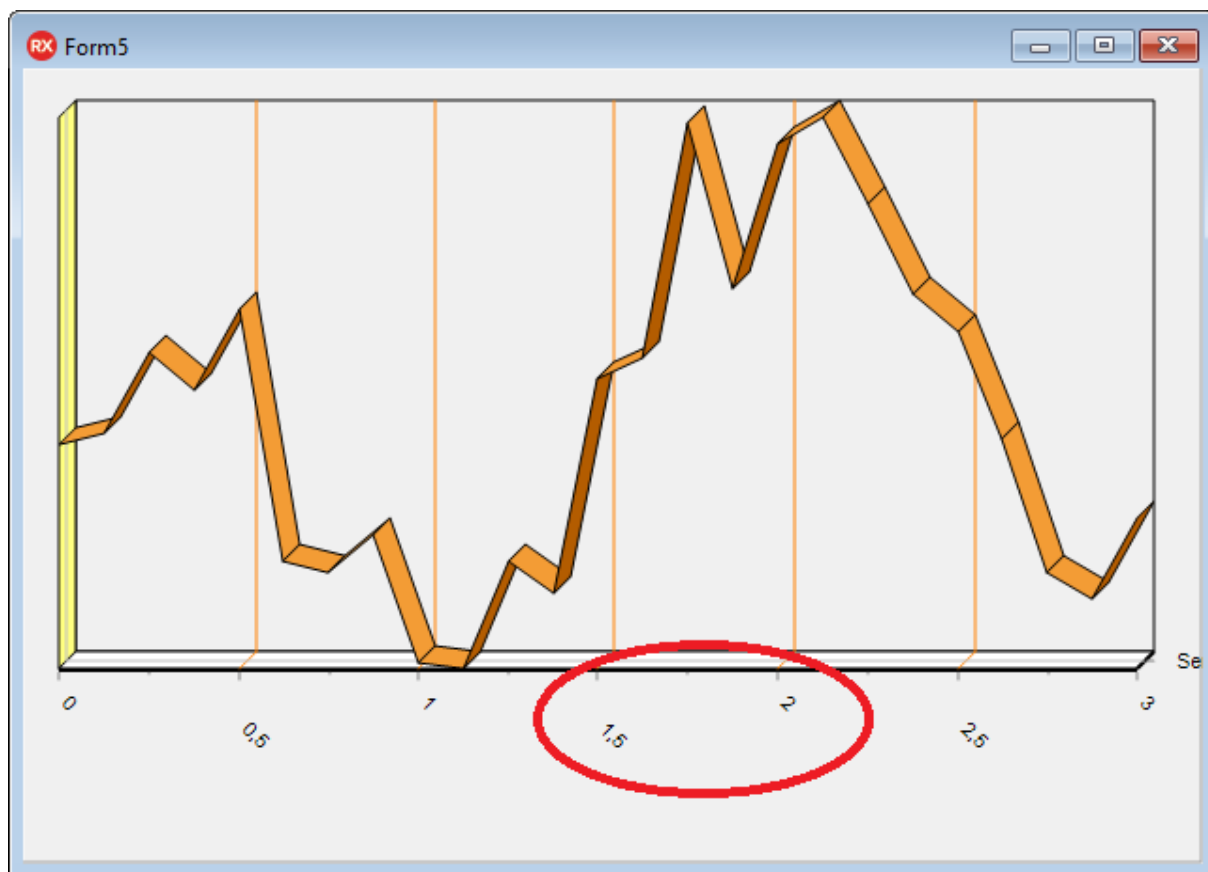
LabelsSize

La propriété LabelsSize vous permet d'agrandir la zone prévue pour les valeurs



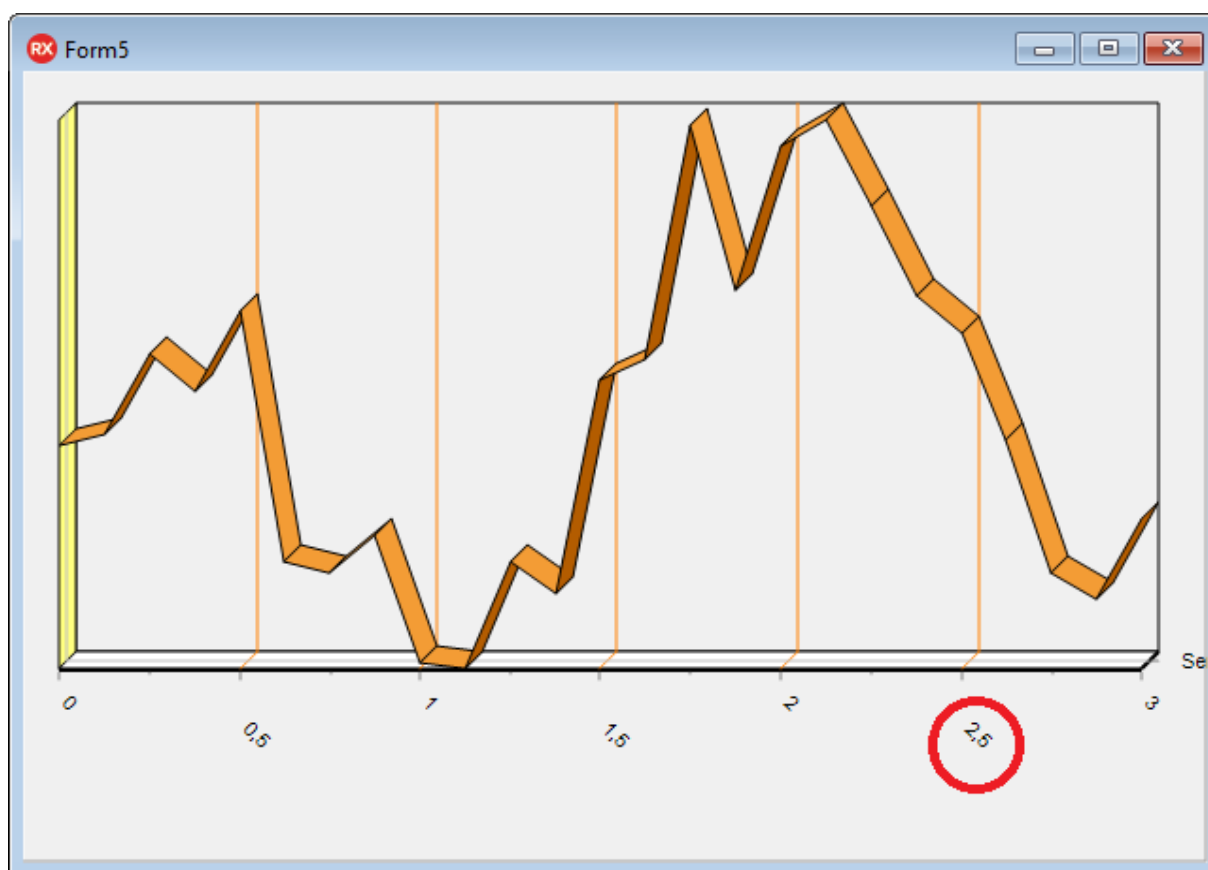
LabelsAlternate

La propriété LabelsAlternate permet une meilleure lecture des axes, une valeur sur deux est décalée



LabelsAngle

CChart.BottomAxis.LabelsAngle permet de modifier l'angle.
Sur l'exemple suivant, l'angle est de 315°



LabelsSeparation

La propriété `LabelsSeparation` spécifie le pourcentage de la distance minimum entre les labels

La valeur 0 supprime le calcul des labels qui se superposent.

Pagination

Utilisez la propriété `CChart.Pages` pour avoir une pagination dans le graphique.

Affectez une valeur à la propriété `CChart.pages.MaxPointsPerPage`.

Interrogez la propriété `CChart.pages.count` pour connaître le nombre de pages au total.

Interrogez/affectez la propriété `CChart.pages.Current` pour paginer.

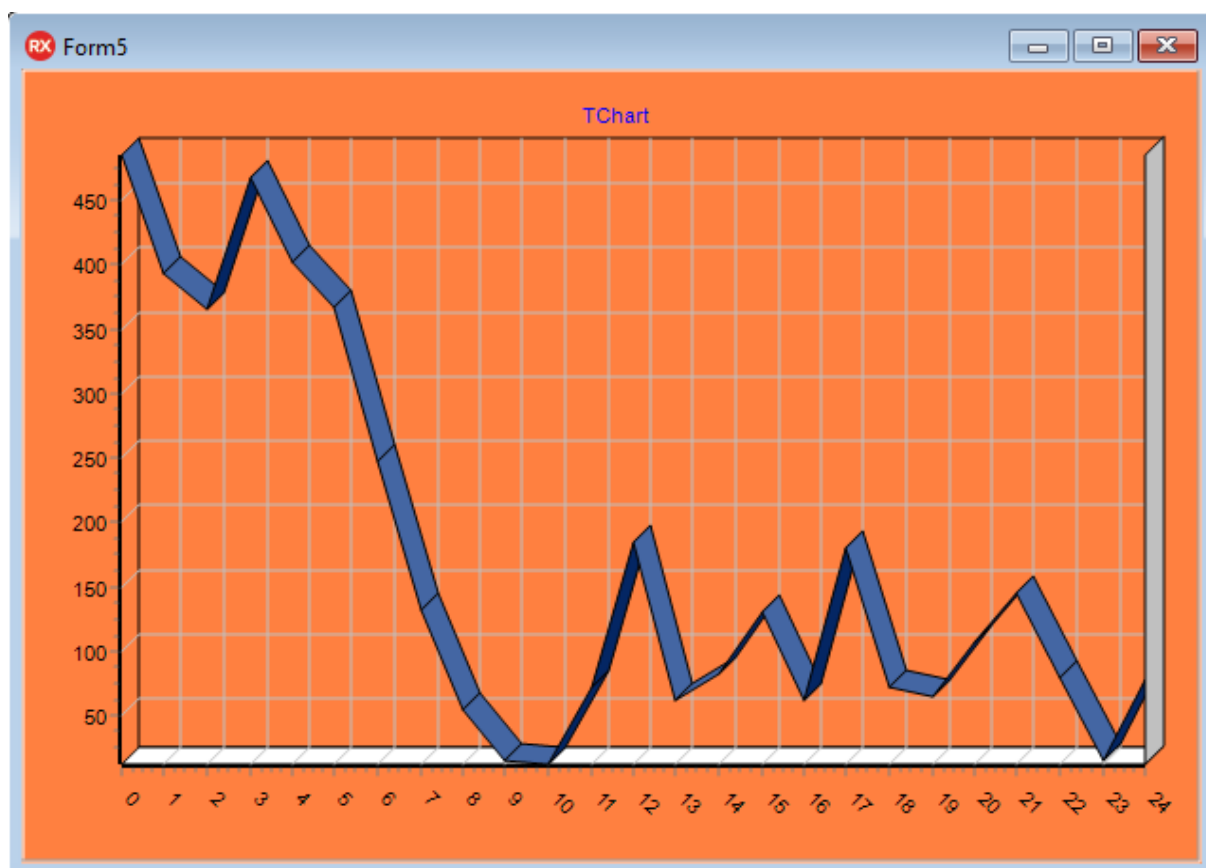
C'est à votre charge d'ajouter des composants qui permettent de naviguer dans la pagination.

Pour réaliser cela, utilisez les propriétés `CChart.pages.count` et `CChart.pages.current`

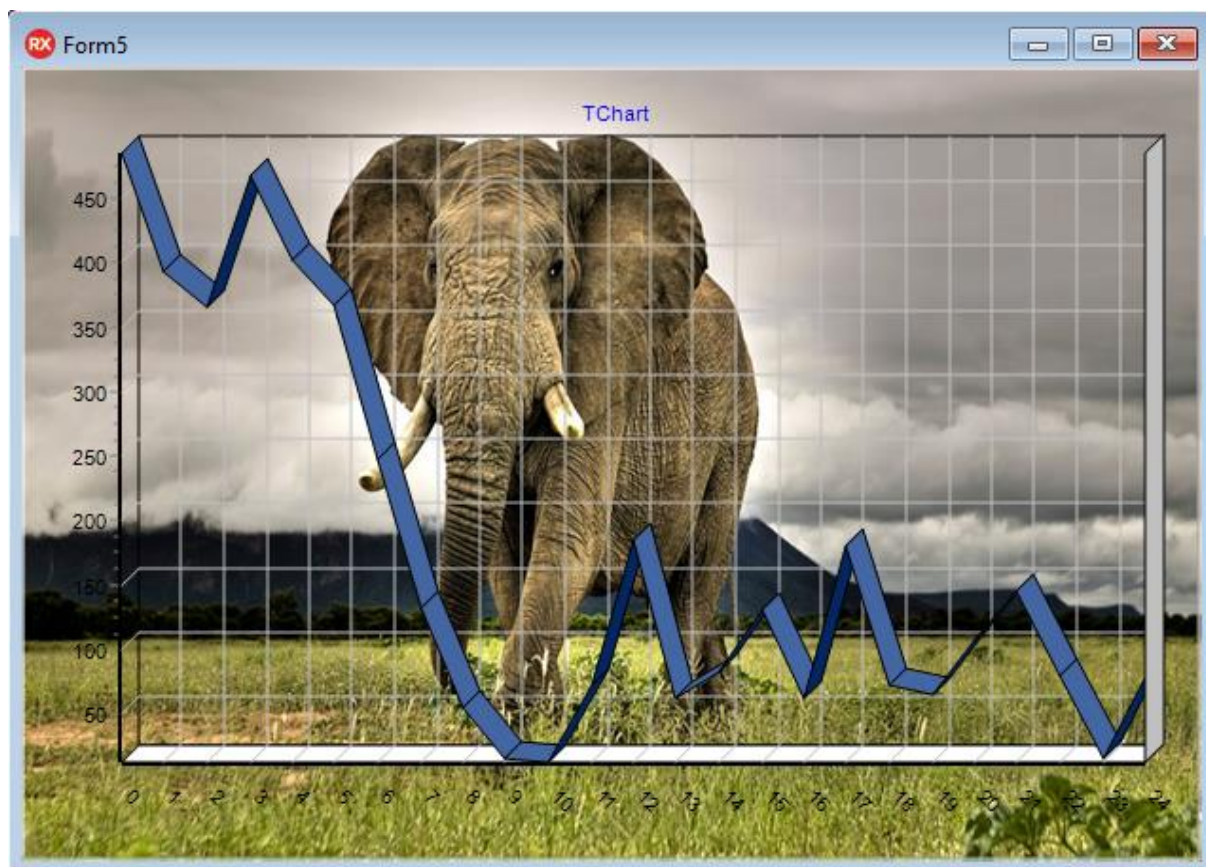
Fond

Le fond du graphique peut être modifié en utilisant

CChart.Color

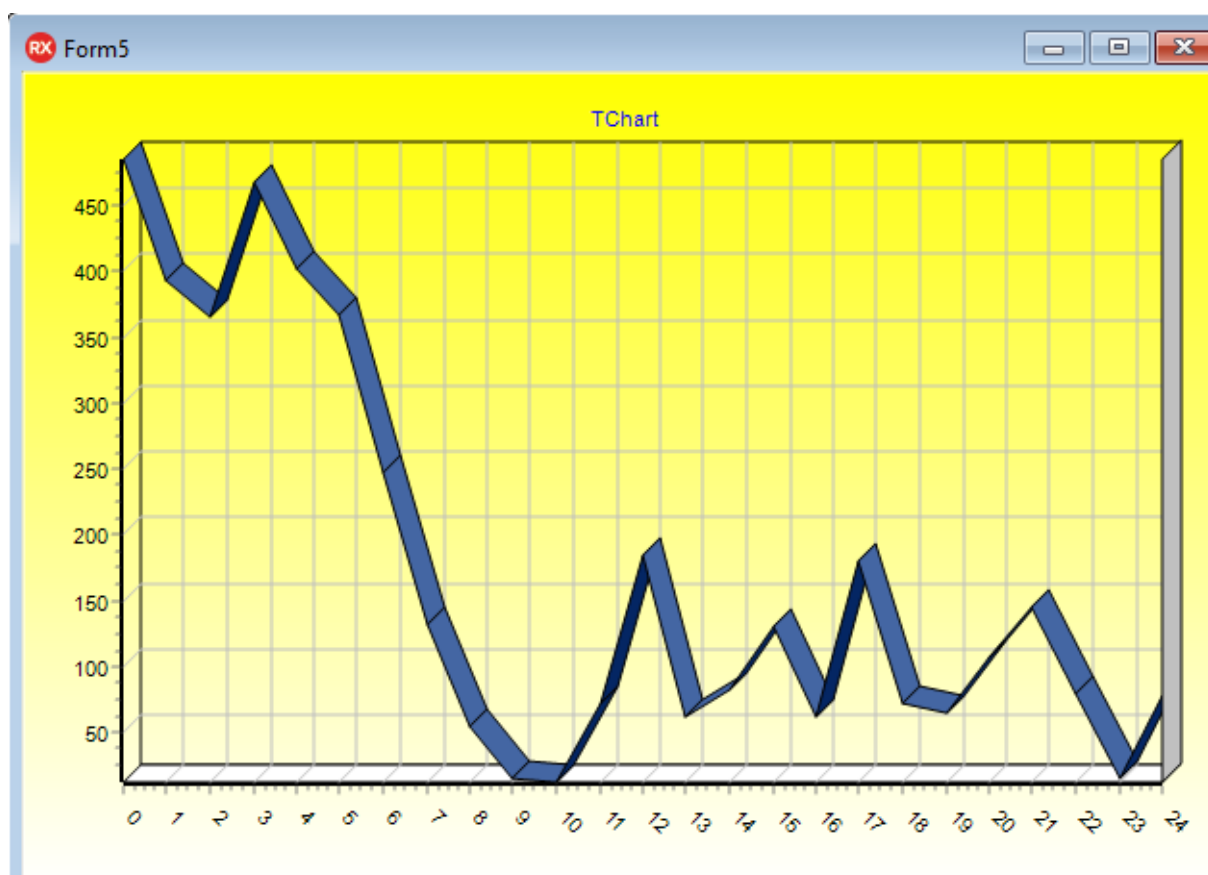


CChart.BackImage



L'aspect de l'image peut être paramétré avec les propriétés **CChart.BackImageInside**, **CChart.BackImageTransp**, **CChart.BackImageMode**

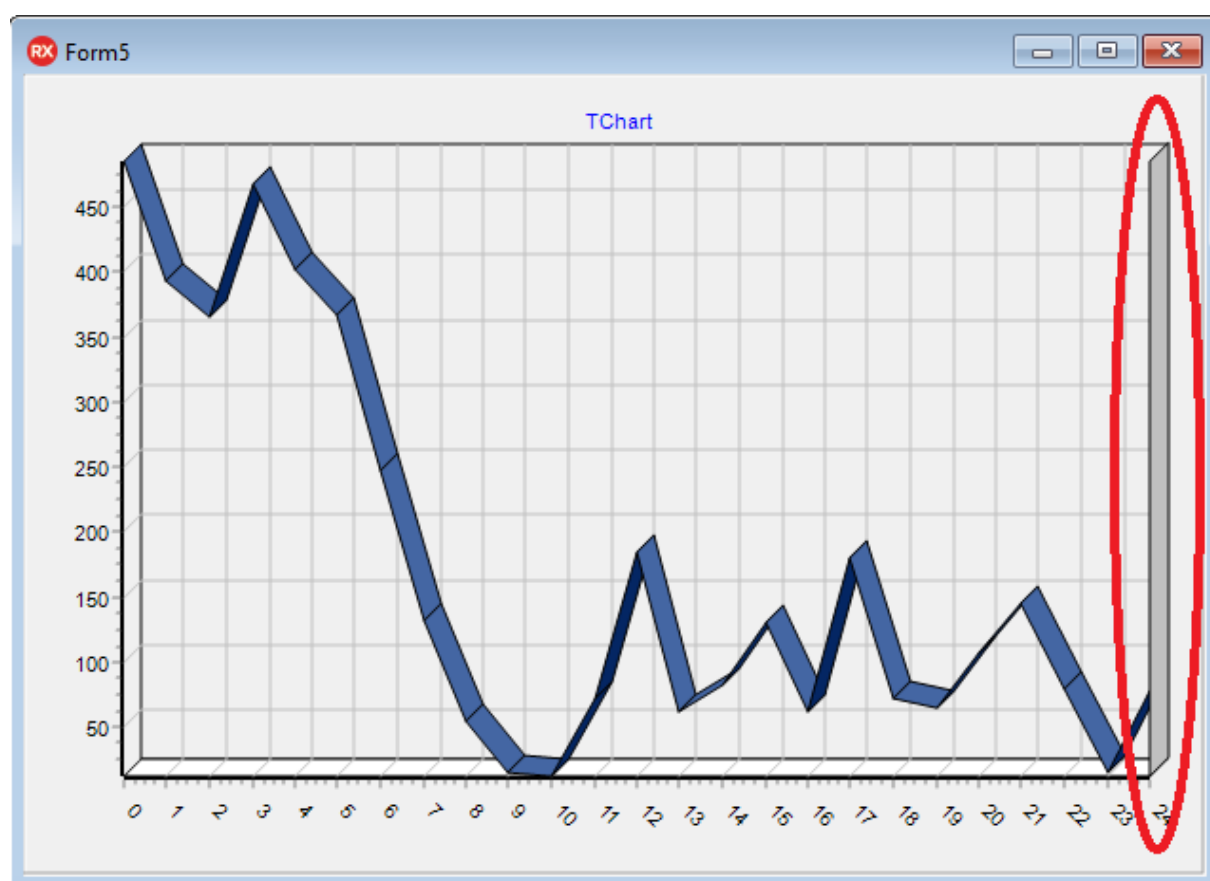
CChart.Gradient



Murs

Il y a quatre walls. LeftWall, RightWall, TopWall et BottomWall.

La figure ci dessous représente le RightWal.



Zoom

Dans le comportement par défaut, l'utilisateur peut zoomer sur une zone du graphique en dessinant un carré de haut en bas et de droite à gauche.

Si l'utilisateur dessine le carré dans un autre sens, le graphique est dézoomé.

L'utilisateur peut ensuite utiliser le click droit pour déplacer la partie zoomée du graphique.

Ce comportement peut être paramétré dans la propriété Zoom du composant CChart.

CChart.Zoom.Allow

Active ou désactive la possibilité de zoomer.

CChart.Zoom.History

Si cette valeur est à true, le mouvement de dezoom reviendra à la valeur de zoom précédente.

Il est alors possible de zoomer deux fois de suite, le dézoom reviendra au premier zoom.

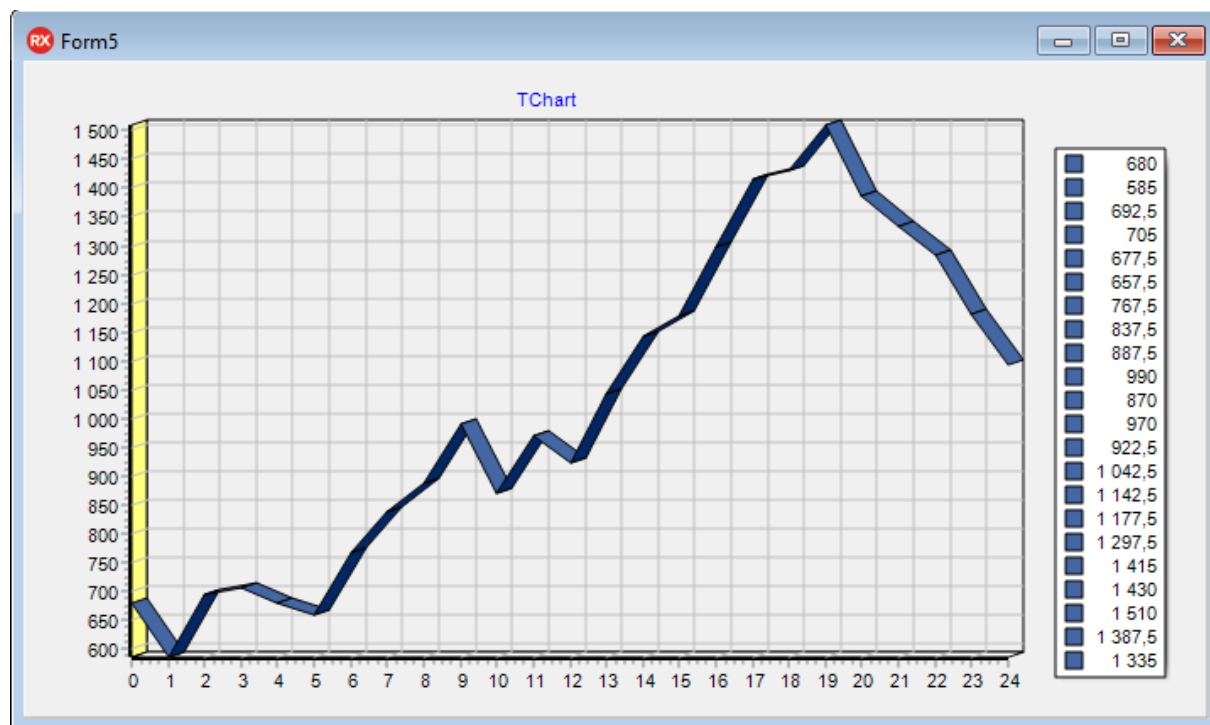
Si cette valeur est à false, le mouvement de dezoom enlèvera complètement le zoom.

3D

Un graphique peut apparaitre en 3D ou non.

Modifiez la propriété **CChart.View3D**

View3D à true



View3D à false :



Les propriétés **CChart.View3DOptions**, **CChart.View3DWalls**, **CChart3DPercent**, **TChartSeries.Dark3D** permettent de paramétrer la vue 3D.

Couleurs

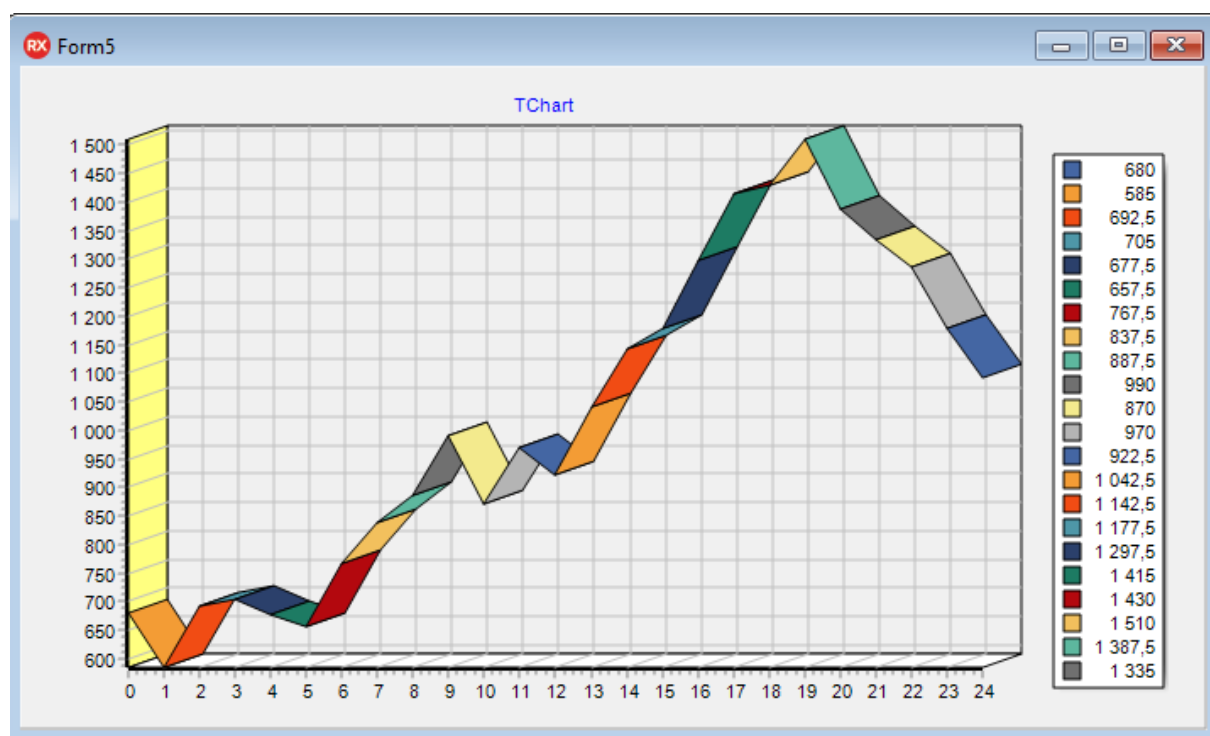
Lors de l'ajout de valeurs dans une série, il est possible de préciser la couleur, à l'aide de paramètres optionnels.

Par exemple :

```
sdAddPie(F1:component:Value:Label:Color)
sdAddBar(F1:component:Value:Label:Color)
sdAddXY(F1:component:ValueX:ValueY:Label:Color)
```

Si aucune couleur n'est précisée, la couleur de la série sera déterminée par la propriété **TChartSeries.SeriesColor**

Il est aussi possible d'affecter true à la propriété **TChartSeries.ColorEachPoint**. Dans ce cas, le composant choisi lui même une couleur pour chaque valeur.



Impression avec graphique

Pour ajouter un graphique dans une impression, utiliser les composants d'impression, CReport et CRepBand.

Posez un composant CRepChart sur un composant CRepBand.

Ajoutez un composant de type CChart.

Modifiez la propriété reference du composant CRepChart en la faisant pointer sur le composant CChart.

Ajoutez les données dans le composant CChart comme vu précédemment, puis lancez l'impression du composant CReport (avec sdPrint ou sdPreview)

Pdf avec graphique.

Pour générer un fichier pdf contenant un graphique, utilisez le composant CReport comme dans l'exemple précédent, puis utilisez la fonction sdSavePdf.

Gdi+

Pour utiliser la bibliothèque GDI+ pour dessiner le graphique, utilisez le composant CChartGDIPlus.

Modifiez la propriété TeePanel du composant CChartGDIPlus, faites la pointer sur le composant CChart.

Modifiez la propriété Active du composant CChartGDIPlus, affectez la valeur true.

Ascenseurs

Le composant ne propose pas d'ascenseurs, mais il est possible de mettre le composant CChart dans un composant CScrollBar et de jouer avec la propriété range de CScrollBar.vertScrollBar et CScrollBar.HorzScrollBar.