

# Reprise Silverdev

I.	Introduction.....	3
A.	Technologie .....	3
B.	Release .....	3
C.	Objets .....	3
D.	Terminologie.....	3
II.	Conversion.....	4
A.	Architecture.....	4
B.	Exemple .....	5
C.	Fenêtre de propriétés.....	12
III.	Appel du programme converti .....	13
A.	Programme CL.....	13
IV.	Compatibilité avec Silverdev Classique .....	14
A.	Appels.....	14
B.	Fenêtres modales .....	18
C.	Appel d'un programme silverdev classique depuis un programme de type handler .....	19
D.	Programme de démarrage .....	19
V.	Améliorations pendant la conversion .....	19
A.	Click sur la croix de fermeture.....	19
B.	Champs.....	19
C.	Désactivation de touches de fonctions ou de champs.....	23
D.	SFL.....	23
VI.	Améliorations après la conversion .....	25
A.	Introduction.....	25
B.	Modification dans le handler .....	26
C.	Modification dans le programme copié .....	41
D.	Modification dans le handler et le programme copié : USERAREA.....	41
VII.	Regénération .....	45
A.	Introduction.....	45
B.	Sauvegarder les propriétés.....	46
C.	Rechargement des propriétés .....	46
D.	Menu Régénérer handler .....	47
E.	Sauvegarde des modifications manuelles .....	49
F.	Suppressions de composants .....	51
G.	Protection .....	52

VIII.	Gérer les évolutions de la version 5250 .....	53
A.	Gérer les évolutions de l'écran d'origine .....	53
B.	Gérer les évolutions du programme d'origine .....	54
IX.	Paramétrage des valeurs par défaut .....	54
A.	Touche de fonction onClose .....	55
B.	Conversions .....	56
C.	SFL .....	56
X.	Contraintes .....	56
XI.	Programmes RPG 3 .....	57
XII.	Solutions de remplacement .....	57
A.	Appel d'un programme système .....	57
B.	Ligne de commande .....	57
C.	Dsply .....	57
D.	Suppression .....	57
E.	RECVF .....	58
XIII.	Champs référence .....	58
XIV.	Licences .....	59
XV.	Plusieurs écrans dans un programme .....	59
XVI.	Erreurs fréquentes .....	59
XVII.	Exemple complet .....	60
A.	Présentation .....	60
B.	Création du contexte .....	60
C.	Génération des handlers .....	61
D.	Copie des programmes originaux .....	62
E.	Lancement du programme converti .....	63
F.	Modifications dans les programmes générés .....	68
XVIII.	Génération multiples .....	70
XIX.	Mots clefs .....	71

# I. Introduction

## A. Technologie

L'outil de reprise de silverdev s'appuie sur la technologie RPGOA.  
Cela impose certaines contraintes, voir le chapitre sur les contraintes.

## B. Release

Du fait de l'utilisation de la technologie RPGOA, le système d'exploitation minimum est V6R1  
L'outil de conversion est présent dans Silverdev à partir de la version V3R8.

## C. Objets

Le programme d'origine contient un programme RPG (\*PGM) et un écran (\*FILE/DSPF)  
La conversion nécessite d'avoir les sources du programme RPG.  
Pour l'écran, seul l'objet suffit.

L'outil Designer de Silverdev permet de générer un handler pour l'écran.  
Un handler est un programme RPG qui sera appelé à chaque fois que le programme principal accèdera à l'écran.  
Un écran silverdev est aussi généré. Cet écran silverdev est utilisé par le programme handler.

Attention : pour ceux qui sont habitués au développement Silverdev classique, l'écran généré pour un handler est différent des écrans que vous connaissez.  
Il contient des fenêtres et des formats.

Le programme d'origine sera copié, puis la déclaration de l'écran sera modifiée dans cette copie.  
L'instruction handler('NOMHANDLER') sera ajoutée dans la déclaration de l'écran.  
Le programme copié sera compilé.

Résumé :

Version 5250	Version Silverdev convertie
Programme d'origine	Programme copié
Ecran 5250	Ecran 5250 Programme handler Ecran Silverdev handler

## D. Terminologie

Dans la suite de ce document, les termes suivants seront utilisés :

Programme d'origine	Source et objet du programme 5550 qui est converti
Programme copié	Source et objet du programme d'origine copié
Programme handler	Source et objet du programme généré qui est utilisé comme handler
Ecran Silverdev handler	Source et objet de l'écran généré. Cet écran généré est utilisé conjointement avec le programme handler.
Ecran 5250	Ecran d'origine. Cet écran doit être présent lors de la compilation et lors de l'exécution du programme

## II. Conversion

### A. Architecture

Pour un programme nommé pgm1 dans la bibliothèque lib1, plusieurs architectures possibles, nous choisirons celle-ci :

Le source du programme d'origine sera copié dans une bibliothèque lib2 et compilé dans lib2.  
Le programme compilé ne sera pas renommé. C'est important car il peut être appelé dans un autre programme converti, cela évite de changer l'appel.

Le programme handler et son source seront générés dans lib2.  
Le programme handler peut prendre le nom de l'écran si celui-ci a un nom différent du programme d'origine. Si l'écran 5250 a le même nom que le programme d'origine, le programme handler devra avoir un autre nom.

L'écran silverdev généré sera généré dans lib2. (source dans l'ifs et objet de type \*USRSPC)

Au final, on aura donc

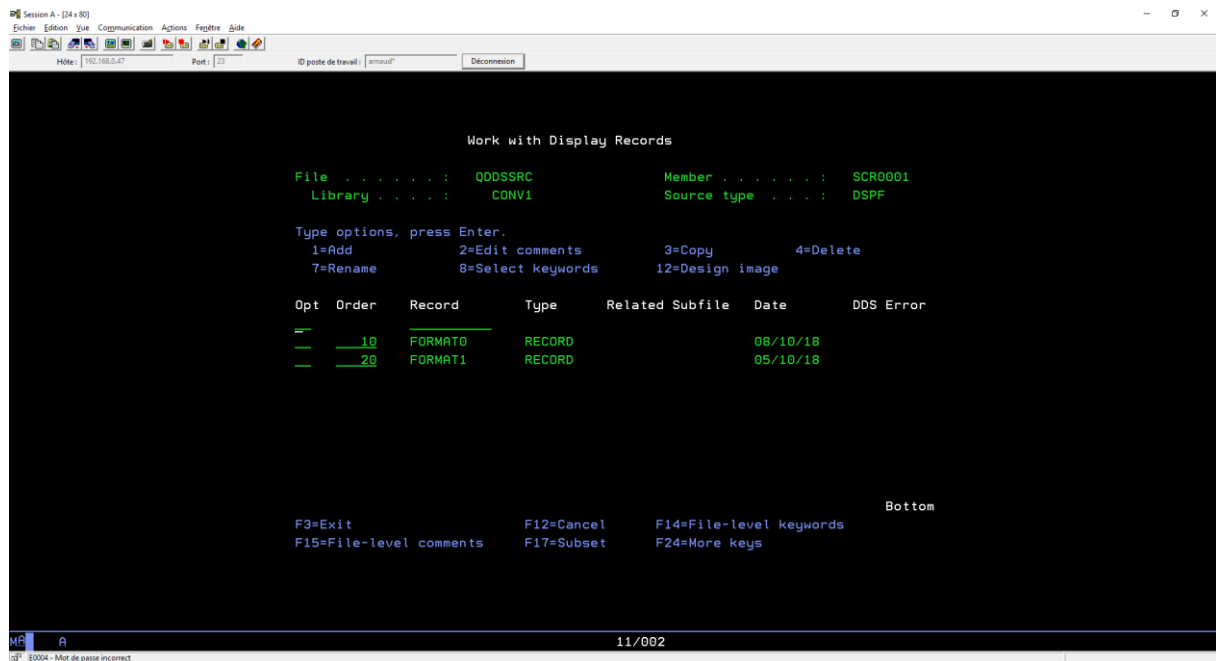
Srclib1	Lib1	Srclib2	Lib2	ifs
Sources de pgm1	Pgm1 (*PGM)	Source de pgm1	Pgm1(*PGM)	Source écran silverdev handler1
	Screen1 (*FILE/DSPF)	Source de handler1	HANDLER1(*PGM)	
			HANDLER1(*USRSPC)	

Remarque 1 : il est possible de ne garder qu'un seul source, en mettant une instruction de pré compilation autour de la déclaration de l'écran.

Remarque 2 : Dans les exemples de ce document, les sources et les objets seront dans la même bibliothèque. srclib1 et srclib2 n'existeront pas

## B. Exemple

Dans l'exemple suivant, nous avons le programme LIBORIGIN/PGM0001 et l'écran LIBORIGIN /SCR0001



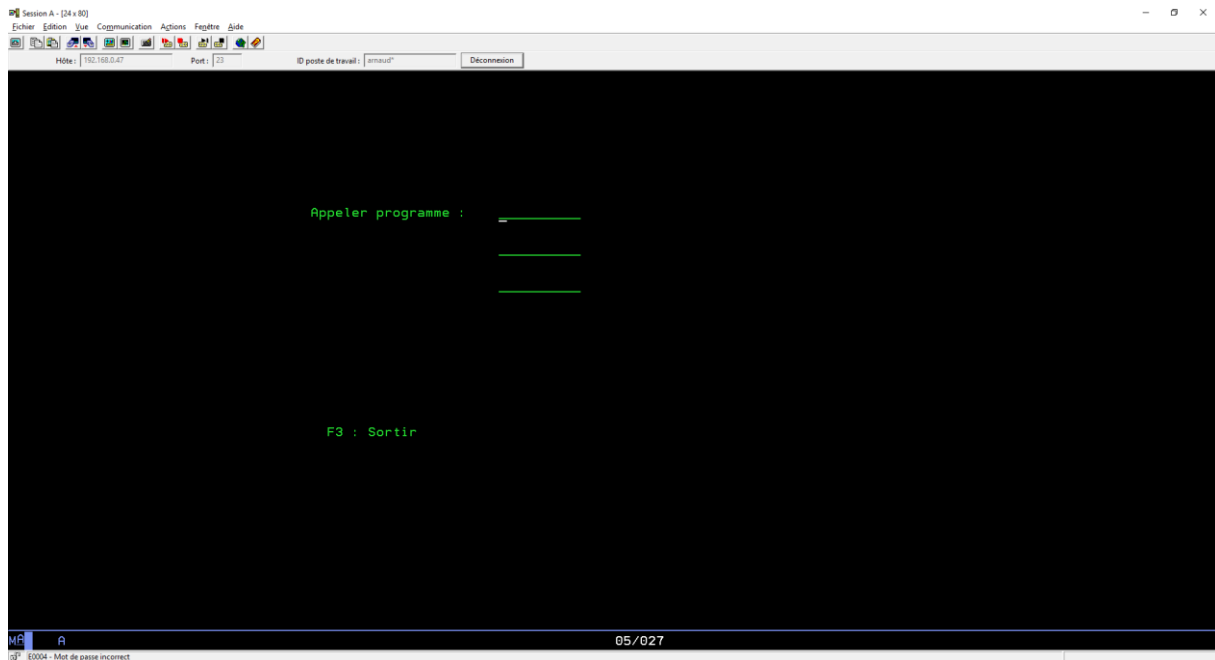
```

FSCR0001  Cf  e          workstn
/copy h,protos

/free
*inlr = *on;
dow  not *in03;
  write format0;
  exfmt format1;
  if *in03 = *off; //touche entrée
    if fld005 <> *blanks;
      monitor;
/end-free
C          call      fld005
/free
  on-error;
  endmon;
  fld005 = *blanks;
  elseif fld006 <> *blanks;
    monitor;
/end-free
C          call      fld006
/free
  on-error;
  endmon;
  elseif FLD008 <> *blanks;
    monitor;
/end-free
C          call      FLD008
/free
  on-error;
  endmon;

  endif;
endif;
enddo;
/end-free

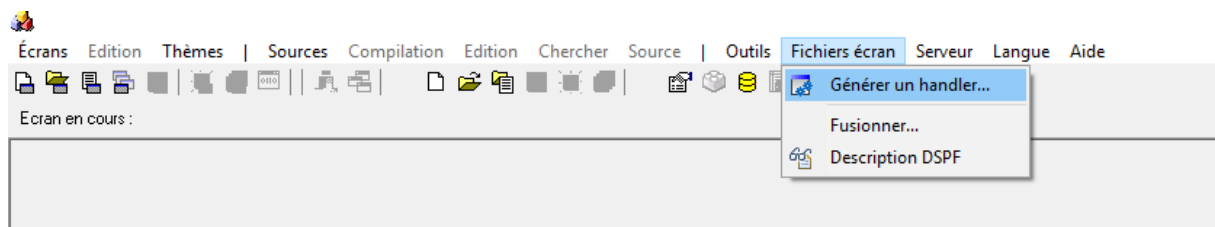
```



Un contexte silverdev est créé dans LIBCONV (voir cours sur silverdev classique pour la création d'un contexte)

Génération du handler :

Utilisez le menu "Fichier Ecran/Générer un handler..."



Renseignez les zones comme indiqué sur la capture d'écran suivante :

**Génération de handler**

Écran 5250

Bibliothèque : LIBORIGIN ...

Nom : SCR0001 ...

Programme Silverdev

Contexte : LIBCONV ...

Programme : HDL0001

Écran : \*PGM

Type : SVDHRPG

Description : Handler pour LIBORIGIN/SCR0001

OK Annuler



**Une fenêtre apparaît, laissez les valeurs pré renseignées, nous reviendrons plus tard sur cette fenêtre.  
Cliquez sur Valider**

Propriétés fenêtre

Fenêtre Silverdev

Largeur :

Hauteur :

Marge :

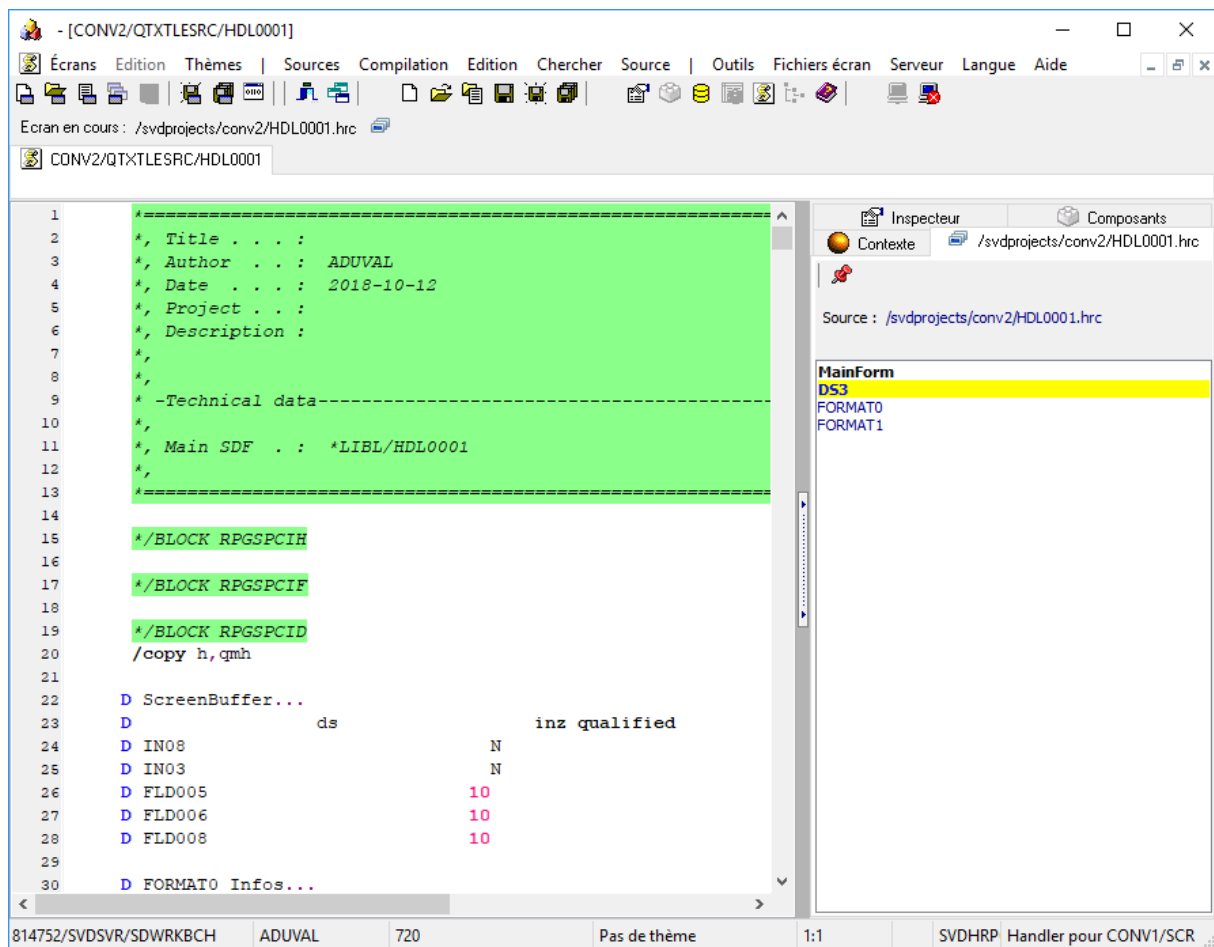
FORMAT0 FORMAT1

Type de format : RECORD

OnClose, touche de fonction :

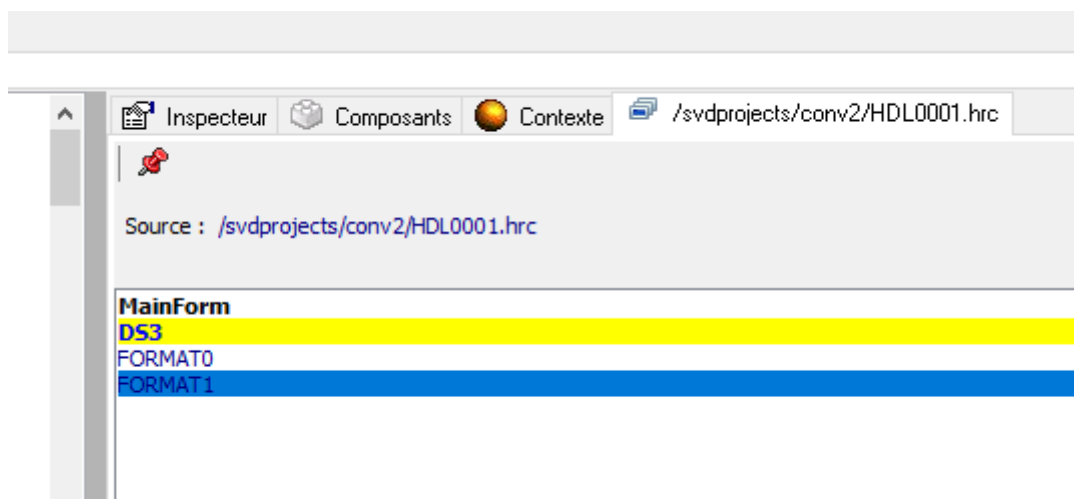
Valider Annuler

Le source rpg généré et le source de l'écran généré s'affichent :

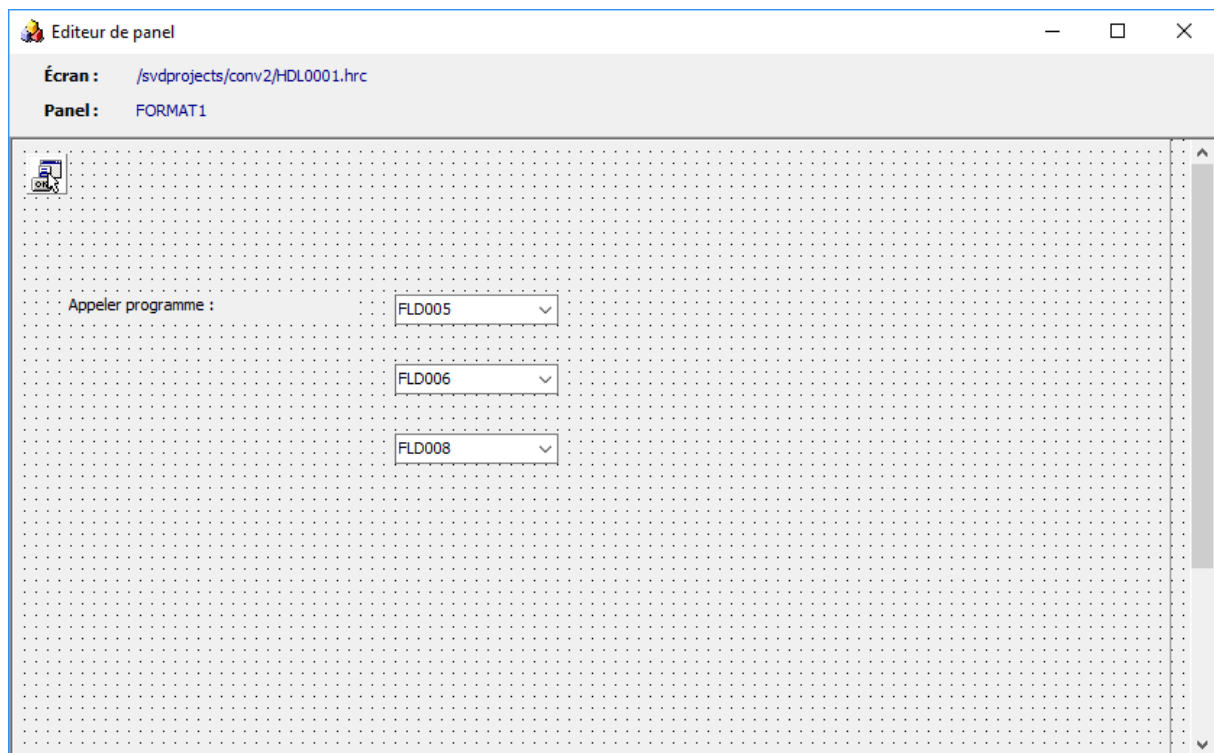


Le programme handler est généré, la suite est facultative.

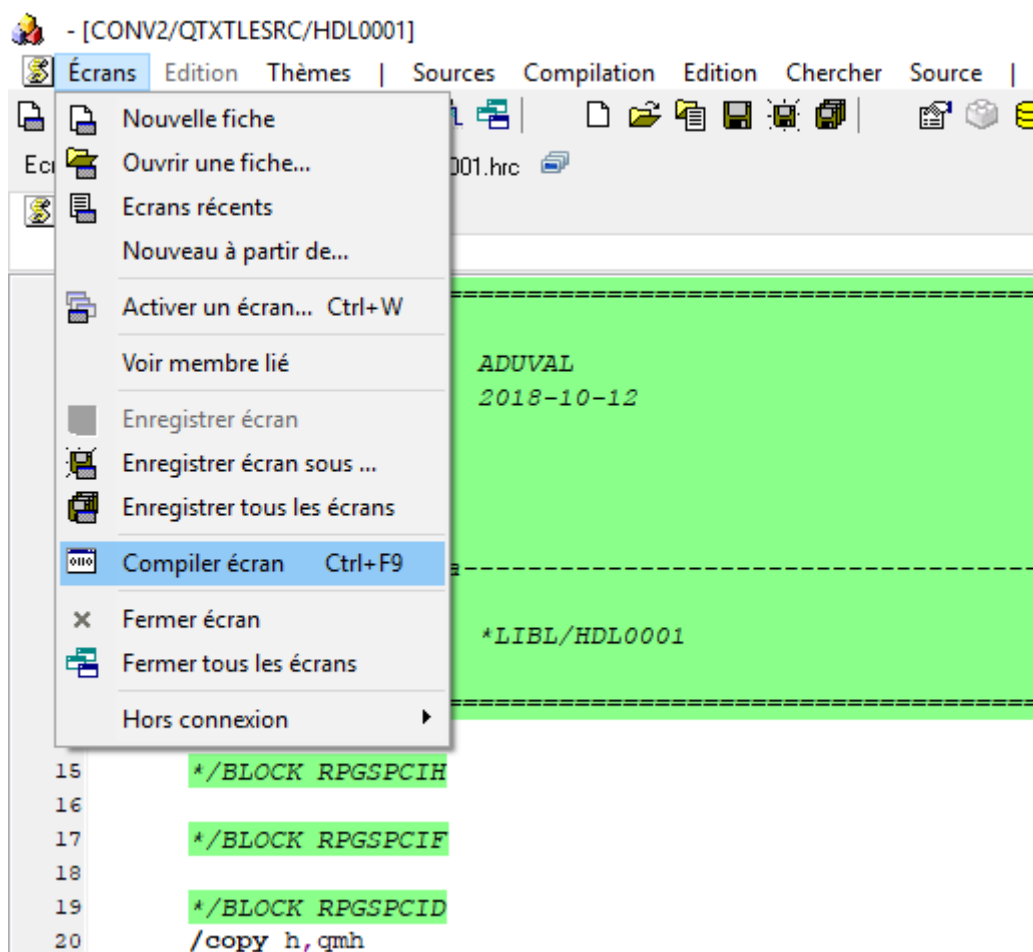
**Figure 1** Pour modifier un format de l'écran silverdev, double cliquez sur son nom



Des modifications peuvent être faites, voir chapitre amélioration des programmes générés



Si vous avez modifié l'écran, vous devez le sauvegarder et le compiler



## C. Fenêtre de propriétés

Lors de la conversion, une fenêtre de propriétés est affichée :

Propriétés fenêtre

Fenêtre Silverdev

Largeur : 300

Hauteur : 600

Marge : 20

FORMAT0 FORMAT1

Type de format : RECORD

OnClose, touche de fonction : FORMAT1.CA03 03

Valider Annuler

La première partie permet de déterminer la hauteur et la largeur de la fenêtre qui servira à afficher les données de cet écran.

Les composants seront placés sur l'écran proportionnellement.

Exemple :

Si un composant est en ligne 2 et colonne 10, pour l'affichage 24\*80, il sera en position :

left :  $10 + 2/24 * 600$

top :  $10 + 10/80 * 800$

Pour chaque format, on trouve un onglet.

On peut dans cet onglet, indiquer quelles sont les touches de fonctions activées lorsque l'utilisateur clique sur la croix de fermeture.

Les touches de fonction F3 et F12 sont pré cochées. Veuillez à les décocher si ces touches de fonction servent à autre chose.

Nous reviendrons sur cette fenêtre de propriétés.

### III. Appel du programme converti

#### A. Programme CL

Nous commençons par créer un cl qui va mettre les bonnes listes de bibliothèques en ligne :

```
pgm
ADDLIBLE LIB(LIBCONV) POSITION(*AFTER QTEMP)
MONMSG MSGID(CPF0000)

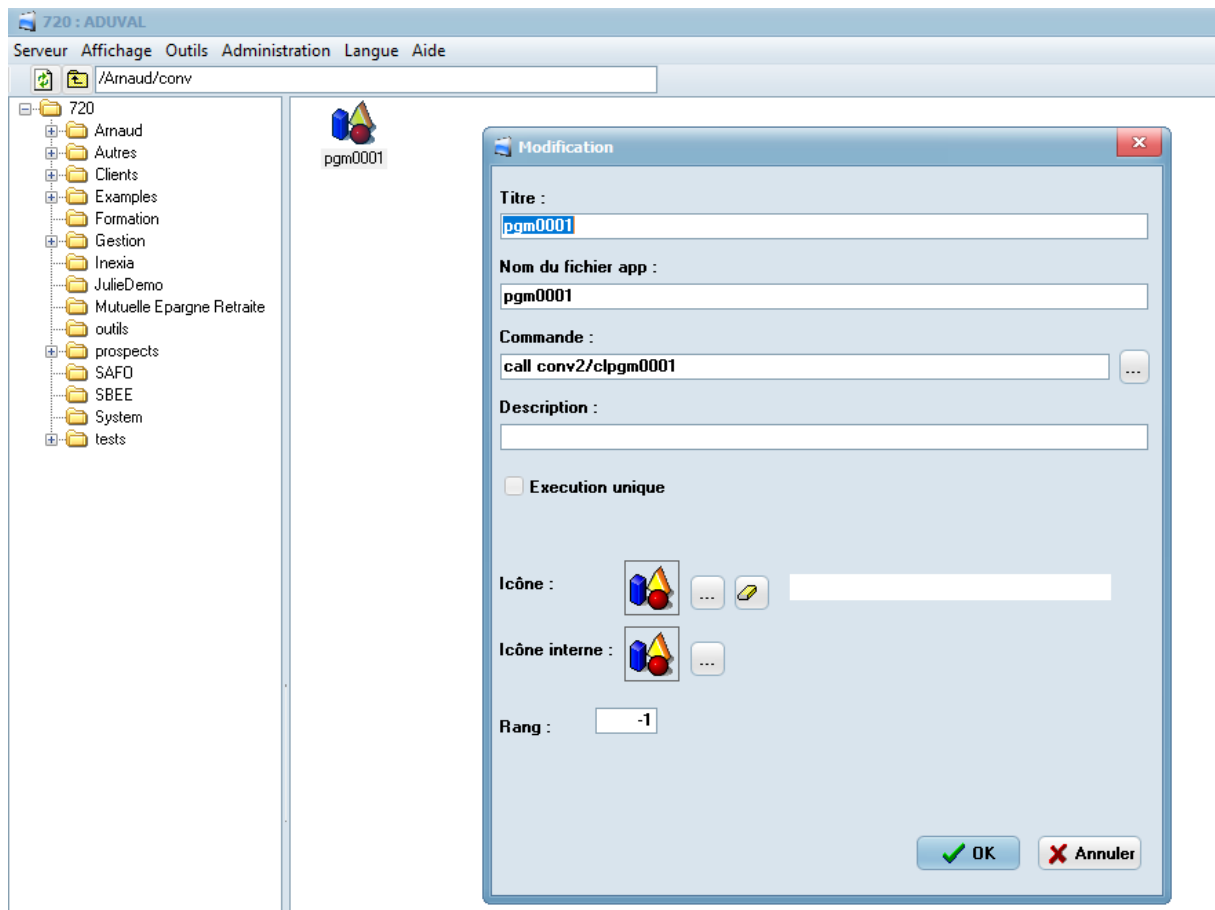
ADDLIBLE LIB(LIBORIGIN) POSITION(*AFTER LIBCONV)
MONMSG MSGID(CPF0000)

CALL PGM(*LIBL/PGM0001)
endpgm
```

La bibliothèque d'origine est mise en ligne car l'écran 5250 doit être en ligne à l'exécution.  
La bibliothèque avec les objets créés est mise en ligne en tête de liste.

Le programme converti est appelé.

Un nouveau raccourci est créé dans MyDesk.



Remarque : MyDesk est le programme de lancement des applications Silverdev.  
Voir le cours général sur Silverdev pour avoir plus d'informations sur MyDesk.

## IV. Compatibilité avec Silverdev Classique

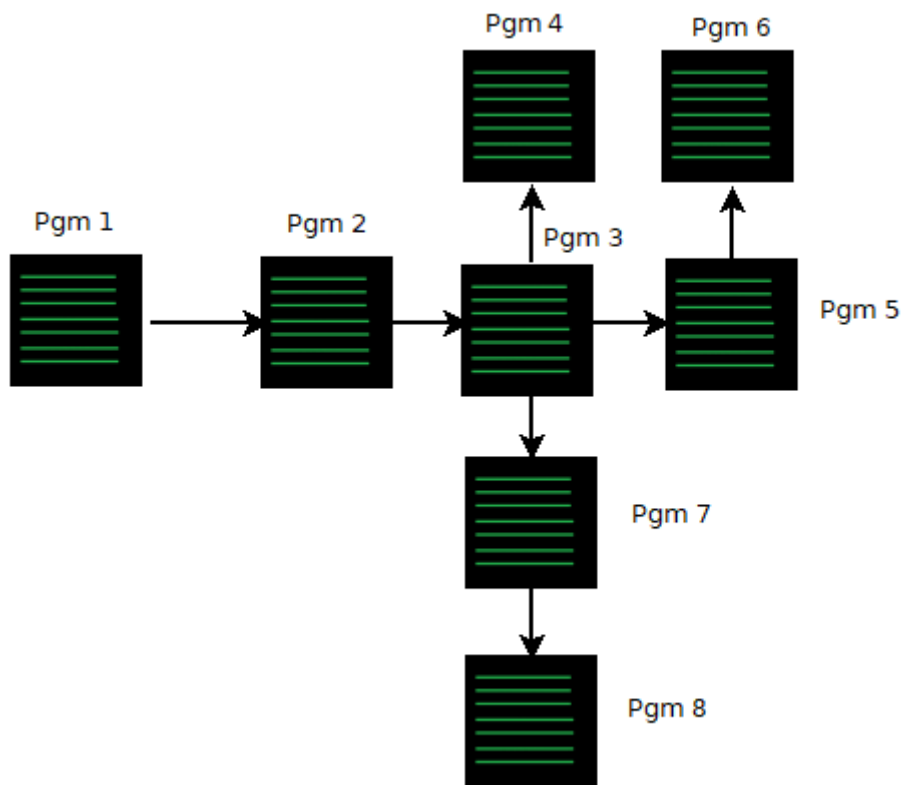
### A. Appels

Un des points forts de l'outil de conversion silverdev, c'est la compatibilité entre les programmes silverdev convertis et les programmes silverdev classiques.

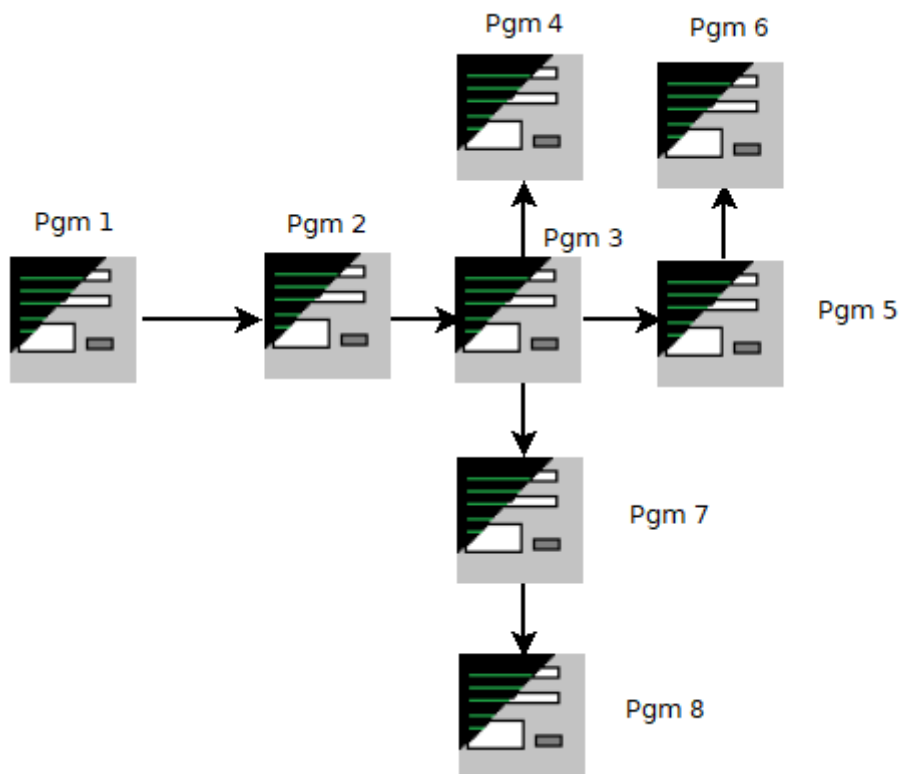
Cela signifie que un programme silverdev classique peut appeler un programme silverdev converti et inversement.

Il est ainsi possible de convertir une chaîne de programmes composant une application et de remplacer progressivement les programmes convertis par des programmes silverdev classique en repensant l'ergonomie.

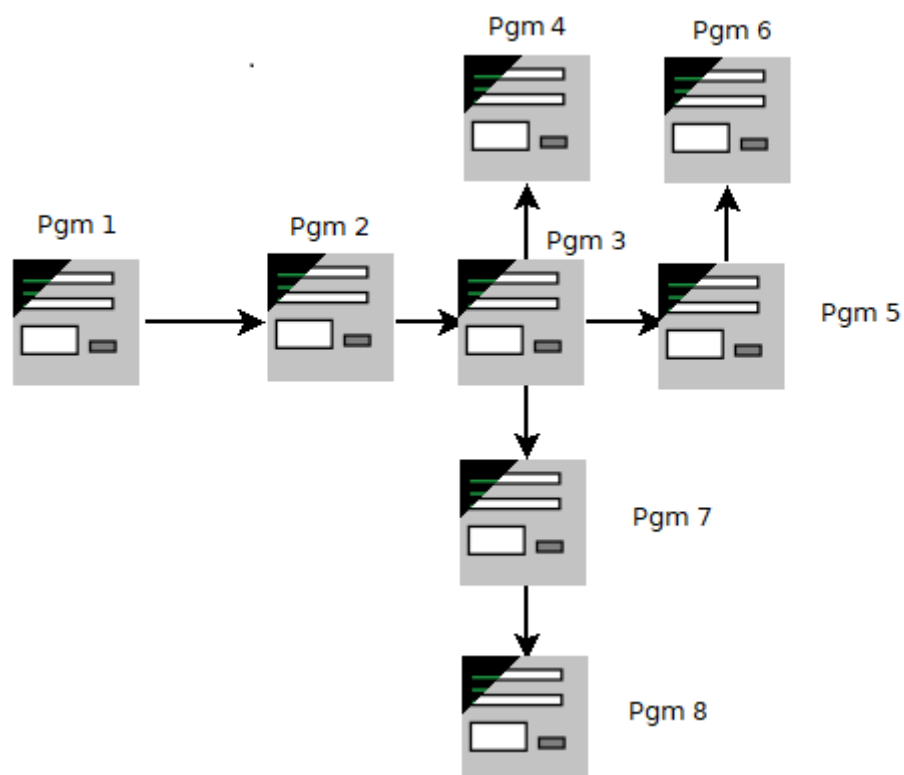
Soit la chaîne de programme suivante :



Après la création des handlers, on obtient la chaine suivante :

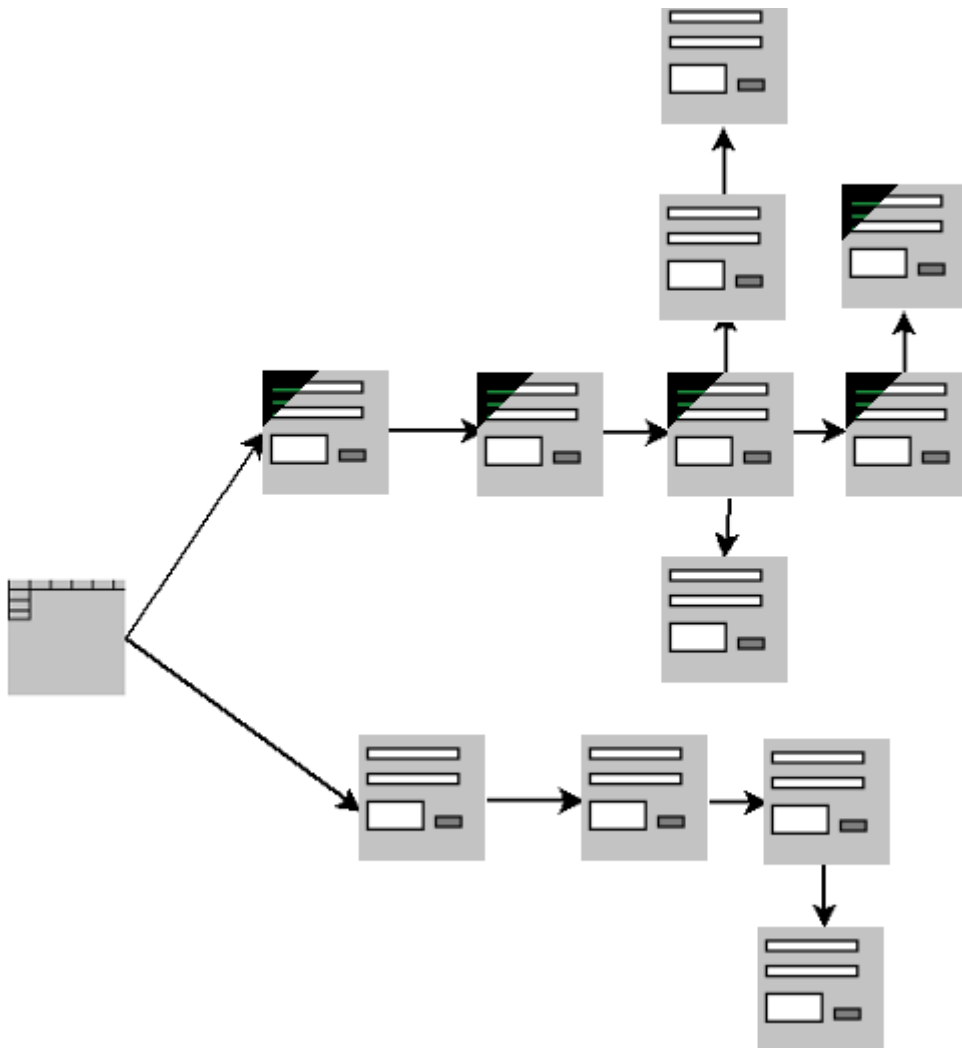


On peut ensuite apporter des améliorations aux programmes générés :



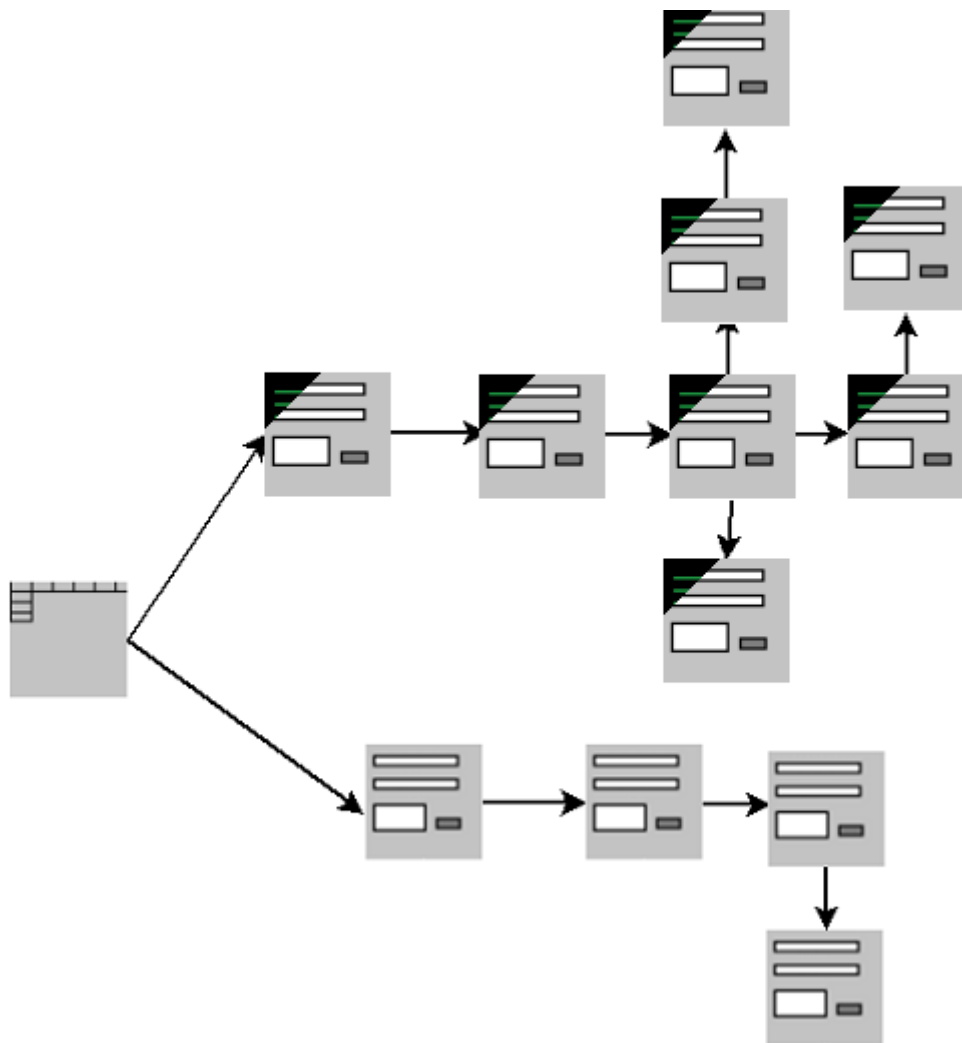
Il est ensuite possible de remplacer certains programmes par des programmes en silverdev classique





Dans l'exemple ci-dessus, les programmes 7 et 8 ont été refait en silverdev classique et ont été fusionnés au passage. Le programme 4 a été refait en silverdev classique, et fait maintenant appel à un nouveau programme, le programme 9 fait en silverdev classique

Il est ensuite intéressant de créer un programme en silverdev classique, à la racine de la chaine de programme, comprenant un menu qui peut appeler des programmes convertis ou des nouveaux programmes silverdev classique.



## B. Fenêtres modales

Afin d'avoir un comportement similaire aux programmes 5250 , mais surtout de respecter l'ergonomie des programmes 5250 d'origine, les programmes silverdev convertis utilisent systématiquement des fenêtres en modal.

Chaque écran 5250 converti a sa propre fenêtre.

Cela permet la compatibilité avec les programmes silverdev classique.

Si par exemple une chaine de programmes est composée de pgm1, pgm2 et pgm3 comme suit :

Programme converti	Programme silverdev classique
Pgm1	
	Pgm2
Pgm3	

il est possible que pgm1 appelle pgm2 qui appelle pgm3.

Cela ne serait pas possible si tous les programmes convertis étaient affichés dans une même fenêtre.

## C. Appel d'un programme silverdev classique depuis un programme de type handler

Il faut que le programme silverdev classique soit en modal, sinon, la fenêtre du programme classique va passer derrière la fenêtre du programme handler.

## D. Programme de démarrage

Afin de faire cohabiter des programmes silverdev classiques et des programmes silverdev convertis, une solution simple est de créer un programme de démarrage avec un menu, en silverdev classique, avec un composant CMainMenu. Depuis ce menu, des programmes classiques ou des programmes convertis peuvent être appelés.

# V. Améliorations pendant la conversion

## A. Click sur la croix de fermeture.

Lors de la génération du handler, une fenêtre est affichée.

Elle permet de sélectionner les touches de fonctions qui sont activées lors du click sur la croix de la fenêtre.

Par défaut, les touches de fonction F3 sont cochées

Cela permet à l'utilisateur de sortir de la fenêtre en faisant F3 comme le permet le programme ou en cliquant sur la croix de fermeture de la fenêtre.

Si aucune des touches de fonction n'est cochée, la croix ne fermera pas la fenêtre.

## B. Champs

### 1. Vue 5250

Un schéma sur la droite affiche (grossièrement) l'écran 5250 afin de repérer les champs.

Le champ sélectionné dans la grille apparaît en jaune sur le schéma.

Vous pouvez aussi cliquer sur le schéma pour sélectionner le champ dans la grille.

CONV1/SCR0175

Fenêtre Silverdev

Largeur :

Hauteur :

Marge :

RECORD1 "DS3"

Type de format : RECORD

Nom	Type	Attribut	Composant	Omettre
NUM8	Signed numeric zone:Both		CDateEdit	<input type="checkbox"/>
NUM6	Signed numeric zone:Both		CDateEdit	<input checked="" type="checkbox"/>
NUM8B	Numeric only 8	Both	CDateEdit	<input type="checkbox"/>
TOTO	Alpha shift character:Both			<input type="checkbox"/>

NUM8  
NUM6  
NUM8B  
TOTO

Valider Annuler

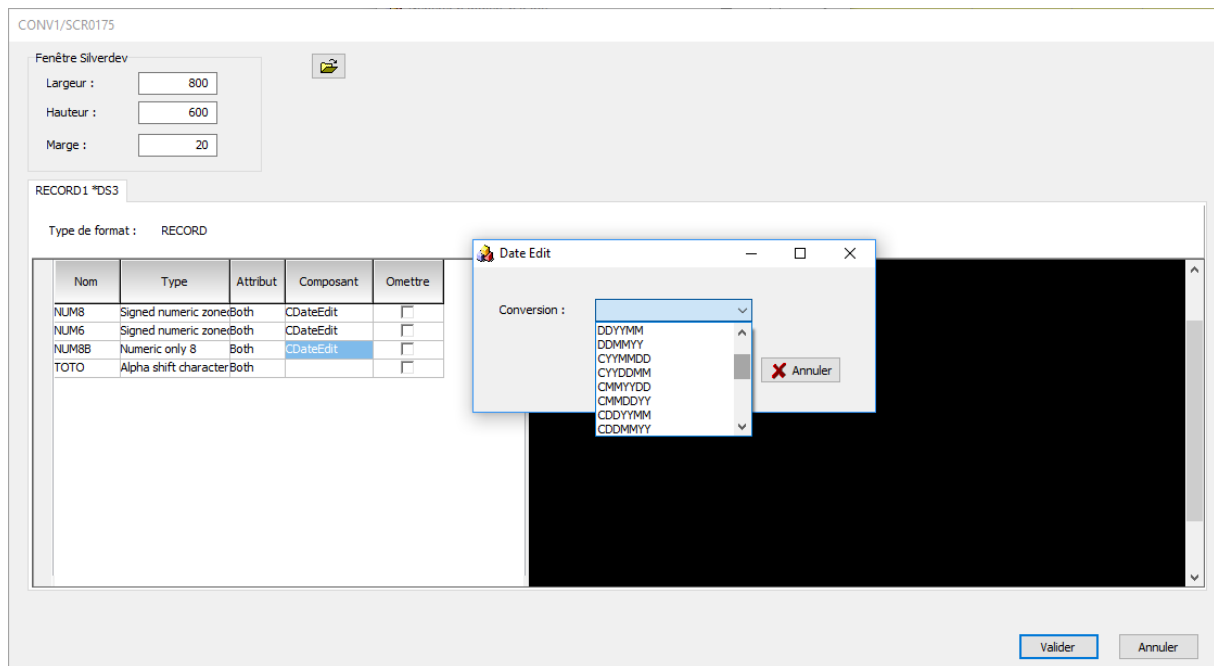
Remarque : si des champs sont superposés et conditionnés par des indicateurs, les deux champs ne sont pas affichés, cependant, le champ sélectionné dans la grille apparaît toujours.

## 2. Type de composant

La colonne "Composant" permet de sélectionner le type de composant qui sera créé pour le champ. Si vous laissez la cellule à vide, le designer choisira automatiquement le type de champ.

## 3. Potentielles dates et heures

Lorsque vous sélectionnez un composant de type date pour un champ numérique, une fenêtre s'ouvre pour que vous donniez le format de la date :



A noter que le composant CDateEdit et le type de conversion peut être proposé automatiquement en configurant le contexte (voir chapitre IX.B)

#### 4. Liste déroulante

Si vous choisissez de générer un composant de type liste déroulante, une fenêtre s'ouvre afin de modifier les propriétés de cette liste déroulante.

ut	Amélioration	Image en-tête
	Liste déroulante	▼

## 5. Cases à cocher

Si vous choisissez "case à cocher" dans la colonne "Amélioration", une boîte de dialogue apparaît pour indiquer les valeurs qui seront récupérées par le programme selon que la case soit cochée ou non.

	Attribut	Amélioration	Largeur	Image en-tête	Libellé	Omettre	Position
acterOutput			202				
acterConstant			135				
acterBoth			202				
acterConstant			135				
acterBoth			7				
acterConstant			135				
acterBoth			67				
acterConstant		Case à cocher	135				
Both			13				
Both			13				
Both			27				
acterConstant			182		(O=Avec, N=	<input checked="" type="checkbox"/>	(7,28) 27
acterConstant			135		Clients AS400	<input type="checkbox"/>	(8,4) 20
acterBoth	Liste déroulante		100			<input type="checkbox"/>	(8,25) 1
acterConstant			175		(O=oui, N=Non	<input checked="" type="checkbox"/>	(8,28) 26

## 6. Omettre des champs

Pour omettre un champ, cochez la case dans la colonne "omettre".  
Les champs dont la case "omettre" est cochée n'apparaissent pas dans le schéma.

Libellé	Omettre	Position
tre de l'imag	<input type="checkbox"/>	(1,15) 30
om client/RS	<input type="checkbox"/>	(4,4) 20
om Client	<input type="checkbox"/>	(4,25) 30
at des clien	<input type="checkbox"/>	(9,4) 20
tat" Client	<input type="checkbox"/>	(9,25) 1
oduit. . . .	<input type="checkbox"/>	(12,4) 20
ode produit	<input type="checkbox"/>	(12,25) 10
ate installat	<input type="checkbox"/>	(18,4) 20
ur Prochai	<input type="checkbox"/>	(18,25) 2
ois Prochai	<input type="checkbox"/>	(18,29) 2
nnée Proch	<input type="checkbox"/>	(18,33) 4
O=Avec, N=	<input checked="" type="checkbox"/>	(7,28) 27
ients AS400	<input type="checkbox"/>	(8,4) 20
	<input type="checkbox"/>	(8,25) 1
O=oui, N=N	<input checked="" type="checkbox"/>	(8,28) 26

àTITPG

Sélections :

Nom client/RS. . . . àKZ001

Cli/Fou/Dis/Par/Tous V

Maintenance . . . . V (O=Avec, N=Sans, rien=tous)

Clients AS400 . . . . S

Etat des clients . . S (F1 pour valeurs)

Région . . . . . SE (F4) Pays : SEL

Produit. . . . . SELPDT (F4)

- code définitif . . S

- Avec maintenance . S

Client rattachement: SELR SNOMRA

Chargé de clientèle: SELPRF

Date installation. . SE SE SELI

Supprimés . . . . . S

A noter que lorsqu'un champ est omis, il n'apparaît plus dans le graph 5250.  
(Sauf s'il est sélectionné dans la grille)

## C. Désactivation de touches de fonctions ou de champs

Parfois, il est souhaitable de désactiver une touche de fonction ou de ne pas afficher un champ.  
Par exemple, si la touche de fonction F9 affiche une ligne de commande, on peut désélectionner cette touche de fonction et enlever le texte affichant "F9 = Ligne de commande"

## D. SFL

Les sous fichiers sont tellement courants en 5250, qu'un panel de propriétés a été ajouté spécialement pour cela.  
Ce panel n'est visible que pour les formats de type sfl.

CONV1/SCR0030

Fenêtre Silverdev

Largeur :

Hauteur :

Marge :

FSFL \*DS3 FCTLSFL \*DS3 EMPTY \*DS3 FOOTER \*DS3

Type de format : SFL

>> Touches de fonction

>> Champs

✓ Sfl

Menu contextuel

☒ Impression

☒ Export

Couleurs grille

Lignes impaires :

Lignes paires :

☒ Alternier les couleurs

Divers

Hauteur entête

☒ Colonne de gauche

Valider Annuler

La case à cocher « impression » rajoute un menu contextuel pour imprimer le contenu de la grille.  
La case à cocher « Export » rajoute un menu contextuel, pour exporter le contenu de la grille.

La zone « couleurs grille » permet de créer une grille dont les lignes ont des couleurs alternées.



CONV1/SCR0219

Fenêtre Silverdev

Largeur :

800

Hauteur :

600

Marge :

20

FORMAT1

Type de format : RECORD

Touche de fonction

OnClose, touche de fonction : CACF03 03

Touche de fonction	Omettre
CACF03	<input type="checkbox"/>
CACF09	<input checked="" type="checkbox"/>
ENTER	<input type="checkbox"/>

Champs

Champ	Conversion numérique	Omettre
FLD001		<input type="checkbox"/>
Valeur saisie :		<input type="checkbox"/>
FLD002		<input type="checkbox"/>
F9 = ligne de commai		<input checked="" type="checkbox"/>

Valider

Annuler

## VI. Améliorations après la conversion

### A. Introduction

L'ensemble programme copié + programme handler suit l'algorithme du programme d'origine. L'ergonomie ne peut donc pas être entièrement modifiée.

Si vous souhaitez repenser entièrement l'ergonomie du programme, il vaut mieux réécrire ce programme en silverdev classique.

Cependant il est possible de modifier le programme généré (programme copié + programme handler) pour apporter quelques améliorations.

25

## B. Modification dans le handler

### 1. Introduction

La modification du handler se fait de la même manière qu'un développement Classique sous Silverdev.

Attention, le nom de la fenêtre est stateInfo.F1, et les noms des composants sont qualifiés.

N'écrivez aucun code entre les balises SvdGenerated tag++ et SvdGenerated tag—, ou votre code serait perdu. (tag représente une expression)

Le code entre ces deux balises est grisé afin de ne pas être tenté de le modifier :

```
*/BLOCK RPGSPCID
*SvdGenerated Declarations++
/copy h,qmh

D setInIndicators...
D          pr
D aFormat          10      value

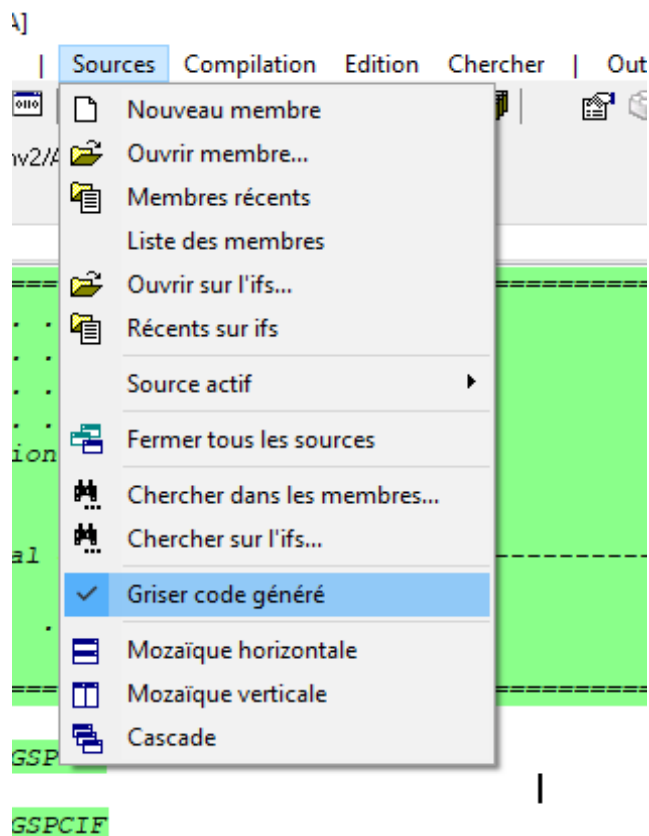
D changeInIndicators...
D          pr
D aInd              2  0  value
D aVldCmdKey        N    value

D DsScreenBuffer...
D          ds          inz qualified
D FLD001            8s 0

D DsFORMAT0_Infos...
D          ds          likeDs(DsRecordInfos) inz

D DsFORMAT0_IN...
D          ds          inz qualified
D FLD001            8s 0
```

Vous pouvez le dé griser avec l'option de menu suivante :



## 2. Modification des propriétés

**Les propriétés des composants générés peuvent être modifiées .**

**Notamment la taille et la position des composants.** Ces modifications peuvent être sauvegardées en cas de régénération du handler. (Voir chapitre re génération)

## 3. Remplacement d'un composant

Un composant généré peut être remplacé par un autre composant. Dans ce cas, vous devez renseigner les moments CUSTOMREAD et CUSTOMWRITE

### a) Exemple 1 :

3 zones numériques sont remplacées par un composant DateEdit :

Pour que la valeur du composant DateEdit soit transmise au buffer de l'écran :

BLOCK CUSTOMREAD					
D	Ds1	ds			qualified
D	year		4	0	
D	month		2	0	
D	day		2	0	
D	all	1	8	0	

#### BLOCK CUSTOMREAD

```
ds1.all = sdGetInt(stateInfo.F1:'DS3.RECORD1_1.DateEdit1':'value');  
StateInfo.RECORD1_In.year = ds1.year;  
StateInfo.RECORD1_In.month = ds1.month;  
StateInfo.RECORD1_In.day = ds1.day;
```

#### \*/BLOCK CUSTOMWRITE

```
ds1.year = StateInfo.ScreenBuffer.year ;  
ds1.month = StateInfo.ScreenBuffer.month ;  
ds1.day = StateInfo.ScreenBuffer.day ;  
sdSetInt(stateInfo.F1:'DS3.RECORD1_1.DateEdit1':'value':ds1.all);
```

Le block CustomTest permet d'ajouter un test sur la valeur lue :

#### \*/BLOCK CUSTOMTEST

```
if StateInfo.RECORD1_In.year = 0;  
row = row +1;  
sdSetCell(StateInfo.fError:'sf11':'text':  
Row:'Date must not be null');  
sdSetCell(StateInfo.fError:'sf11':'control':Row:  
sdGetFormName(StateInfo.F1)+'.' +  
'DS3.RECORD1_1.DateEdit1');  
endif;
```

#### b) Exemple 2 :

L'écran 5250 se présente comme ça :

1 Option 1  
2 Option 2  
3 Option 3  
4 Option 4  
5 Option 5  
6 Option 6  
7 Option 7

Votre choix :     —

F3=Fin

Sélectionnez 'CRadioButton ' comme composant, saisissez la valeur 1,2,3.. dans la propriété Key puis enlevez le champ « Votre choix »

RECORD1 "DS3

Type de format : RECORD

Nom	Type	Attribut	Composant	Omettre
const_1	Alpha shift character	Constant	CRadioButton	<input type="checkbox"/>
const_2	Alpha shift character	Constant	CRadioButton	<input type="checkbox"/>
const_3	Alpha shift character	Constant	CRadioButton	<input type="checkbox"/>
const_4	Alpha shift character	Constant	CRadioButton	<input type="checkbox"/>
const_5	Alpha shift character	Constant	CRadioButton	<input type="checkbox"/>
const_6	Alpha shift character	Constant	CRadioButton	<input type="checkbox"/>
const_7	Alpha shift character	Constant	CRadioButton	<input type="checkbox"/>
const_8	Alpha shift character	Constant		<input checked="" type="checkbox"/>
CHX_1	Alpha shift character	Both		<input checked="" type="checkbox"/>

1 Option 1  
 2 Option 2  
 3 Option 3  
 4 Option 4  
 5 Option 5  
 6 Option 6  
 7 Option 7

Votre choix : CHX

Dans le block customRead, saisissez le code suivant :

```
*/BLOCK CUSTOMREAD
stateInfo.RECORD1_IN.FLD001 =
sdGetInt(stateInfo.F1:'DS3.RECORD1_1.CHX_1':'keyChecked');
```

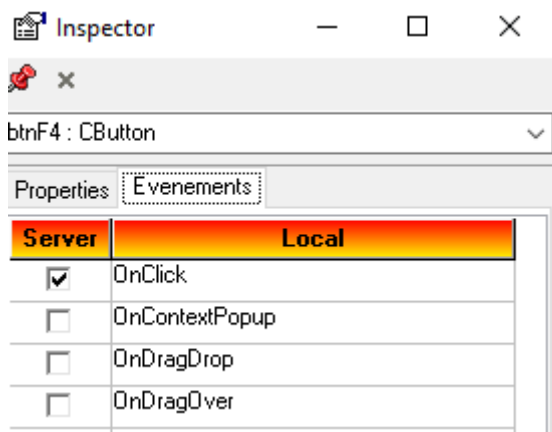
```
*/BLOCK CUSTOMWRITE
sdSetInt(stateInfo.F1:'DS3.RECORD1_1.CHX_1':'keyChecked':
stateInfo.ScreenBuffer.FLD001);
```

#### 4. Ajout d'un bouton.

Dans le programme , la touche F4 permet de faire apparaitre une liste de selection.  
 Nous allons ajouter un bouton qui permettra d'afficher cette liste.



Sur le bouton, cochez l'évènement OnClick,



, puis double cliquez sur l'évènement on click

Saisissez le code qui doit s'exécuter lors du click sur le bouton.

On appelle la fonction CACF04\_ONEXECUTE, mais avant on remet le focus dans la zone concernée, car la fonction CACF04\_ONEXECUTE teste sûrement la zone ayant le focus :

```

*/EVENT DS3_MAT4EF1_1_btnF4_OnClick
* -----
--*
* Description :
* -----
--*
D Parameters      ds          based(pevtinf)
D Win              5u 0
D Evt              48
/free
sdSetFocus(StateInfo.F1: 'DS3.MAT4EF1_1.SAIPGM_1') ;
CACF04_ONEXECUTE() ;
/end-free

```

## 5. Ajout d'une image

L'ajout d'une image dans l'écran est possible, il suffit d'ajouter un composant CImage dans le panel correspondant au format. L'image sera ajoutée à l'écran lorsque le panel sera écrit.

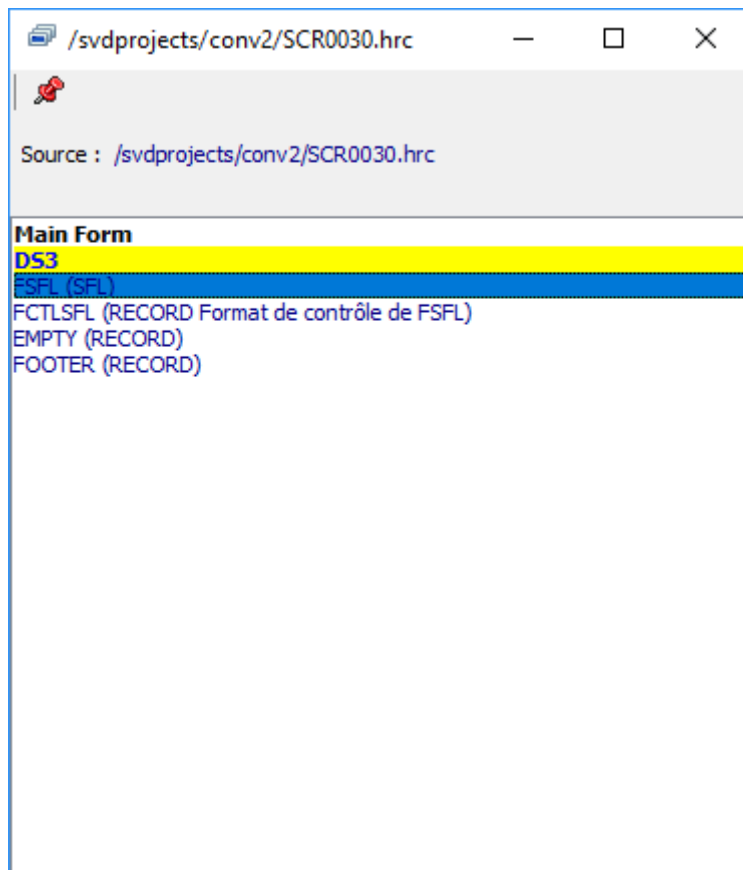
Les composants peuvent être ajoutés directement sur la fenêtre principale ou sur les formats

## 6. Ajout d'un double click sur le sfl

Si dans la version 5250, il est possible de sélectionner une ligne en saisissant une option, vous pouvez ajouter la possibilité d'effectuer un double click pour effectuer cette sélection plus rapidement.

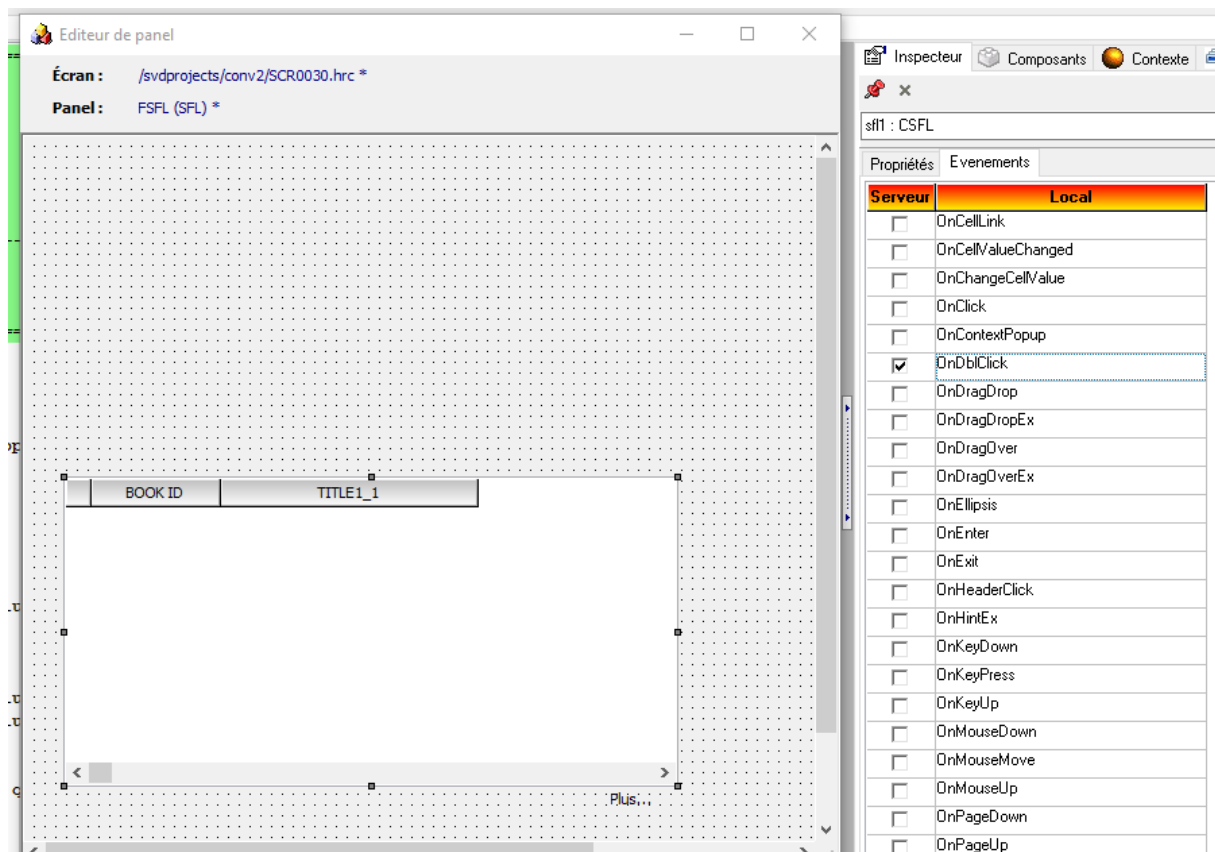
**a) Ouverture du panel**

Dans l'écran généré, double cliquez sur le format correspondant au sfl :



**b) Double click**

Sélectionnez le composant CSFL, puis dans l'inspecteur, cochez l'évènement OnDbfClick :



### c) Code Rpg

Double cliquez sur la ligne "OnDbClick"

Le curseur est inséré dans le code rpg :

```

3069
3070
3071
3072  */EVENT DS3_FSFL_sf11_OnDbClick
3073  * -----*
3074  * Description :
3075  * -----*
3076  D Parameters      ds          based(pevtinf)
3077  D Win              5u 0
3078  D Evt              48
3079  /free
3080
3081  /end-free

```

Entrez le code suivant :



```

p DS3_FSFL_sf11_OnDb1Click...
p                                     B
D                                     pi
D PevtInf                               *   const options(*nopass)
* -----*
* Description :
* -----*
D Parameters          ds                      based(pevtinf)
D Win                                5u 0
D Evt                                48
D row                    s                    10i 0
/free
  row = sdGetInt(stateInfo.F1:'DS3_FSFL_sf11':'rowSelected');
  if row > 0;
    sdSetCell(stateInfo.F1:'DS3_FSFL_sf11':'OPTION1':row:'1');
    Enter_onExecute(pEvtInf);
  endif;
/end-free
p DS3_FSFL_sf11_OnDb1Click...
p                                     E

```

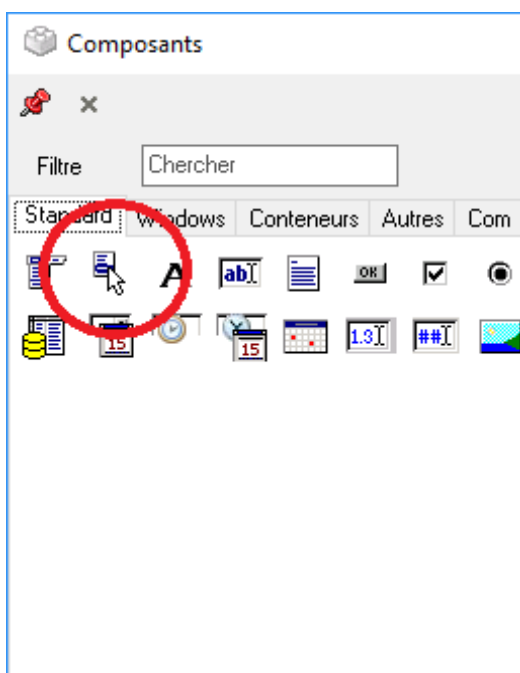
Cela va insérer la valeur '1' dans la colonne Option, puis simuler la touche entrée

## 7. Ajout d'un menu contextuel sur le sfl

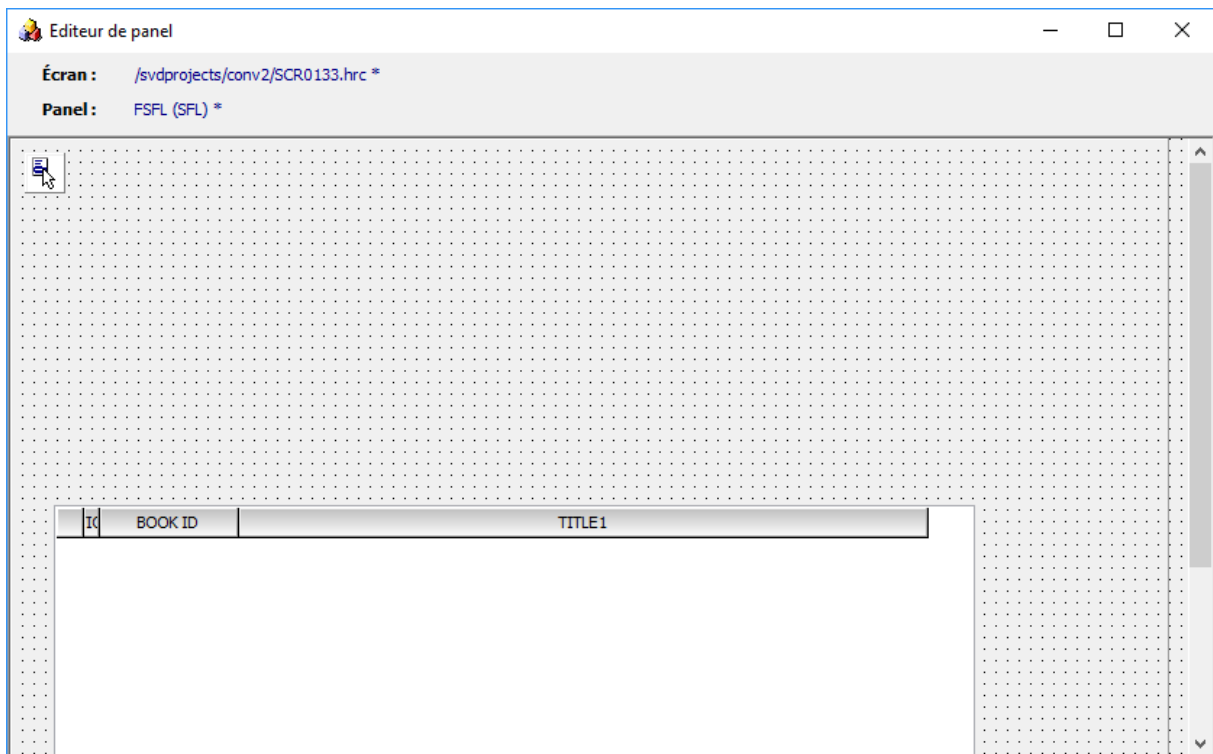
Si plusieurs options sont possibles pour une ligne de sous fichier, il est intéressant d'ajouter un menu contextuel qui simplifiera la saisie de ces options.

### a) Composant CPopupMenu

Sélectionnez dans la palette de composants le composant CPopupMenu



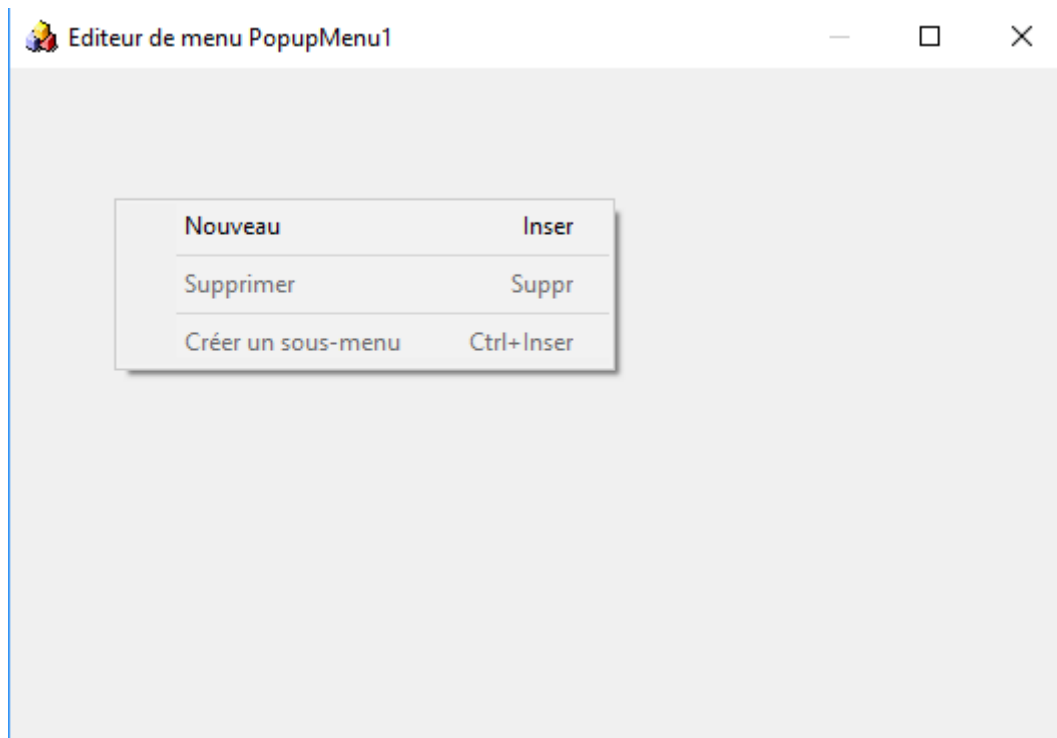
Déposer le composant sur le panel contenant le composant CSfl



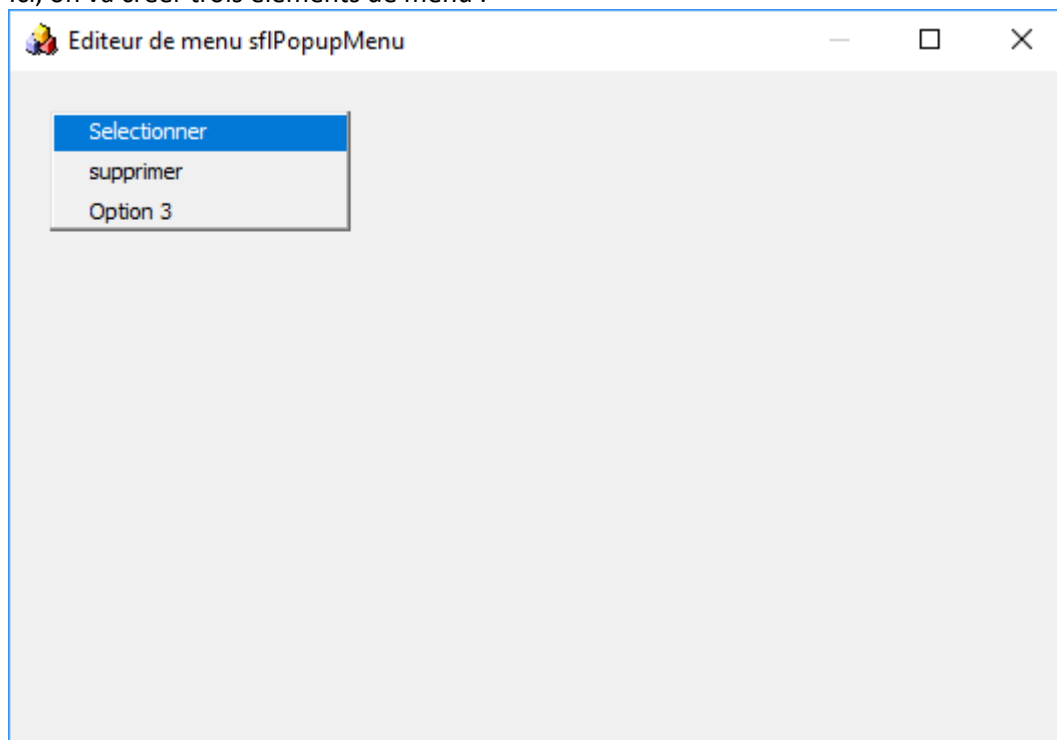
## b) Ajout des éléments

Double cliquez sur le composant

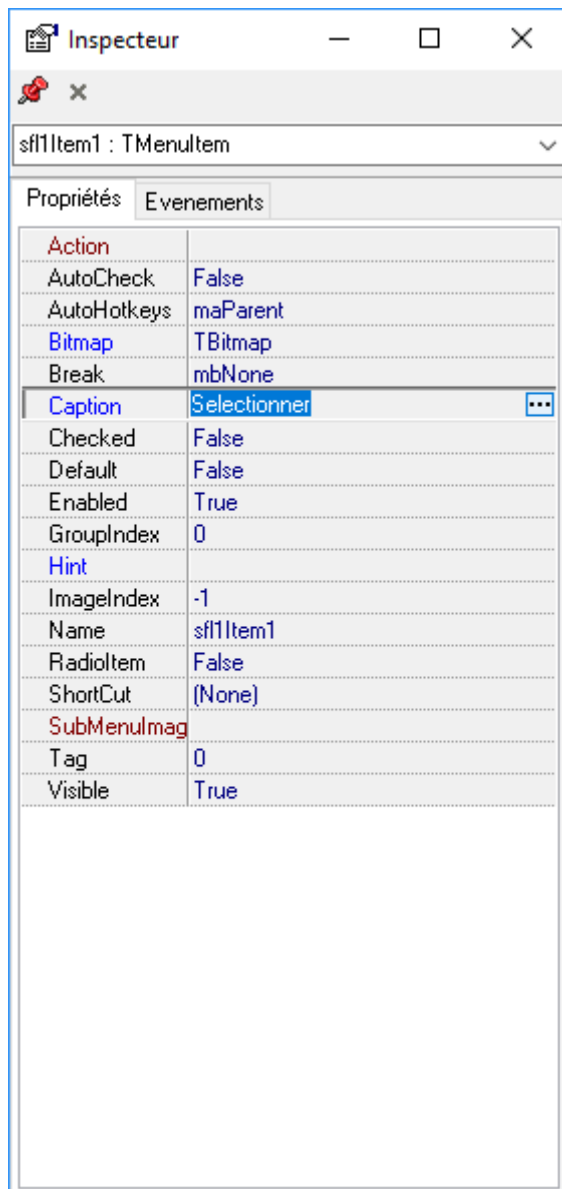
L'éditeur de menu s'ouvre, faites un click droit dans l'éditeur, et sélectionnez "Nouveau"



Ici, on va créer trois éléments de menu :



Les propriétés tels que le texte affiché (propriété caption) peuvent être modifiées dans l'inspecteur de propriétés :



### c) Evènements OnClick

Pour chaque élément, cochez l'évènement onClick, puis saisissez le code correspondant :

```

*/EVENT DS3_FSFL_sf11Item2_OnClick
* -----*
* Description :
* -----*
D Parameters      ds          based(pevtinf)
D Win              5u 0
D Evt              48
D row              s          10i 0
/free
row = sdGetInt(stateInfo.F1:'DS3_FSFL_sf11':'rowSelected');
if row > 0;
    sdSetCell(stateInfo.F1:'DS3_FSFL_sf11':'OPTION1':row:'2');
    Enter_onExecute(pEvtInf);
endif;

/end-free

```

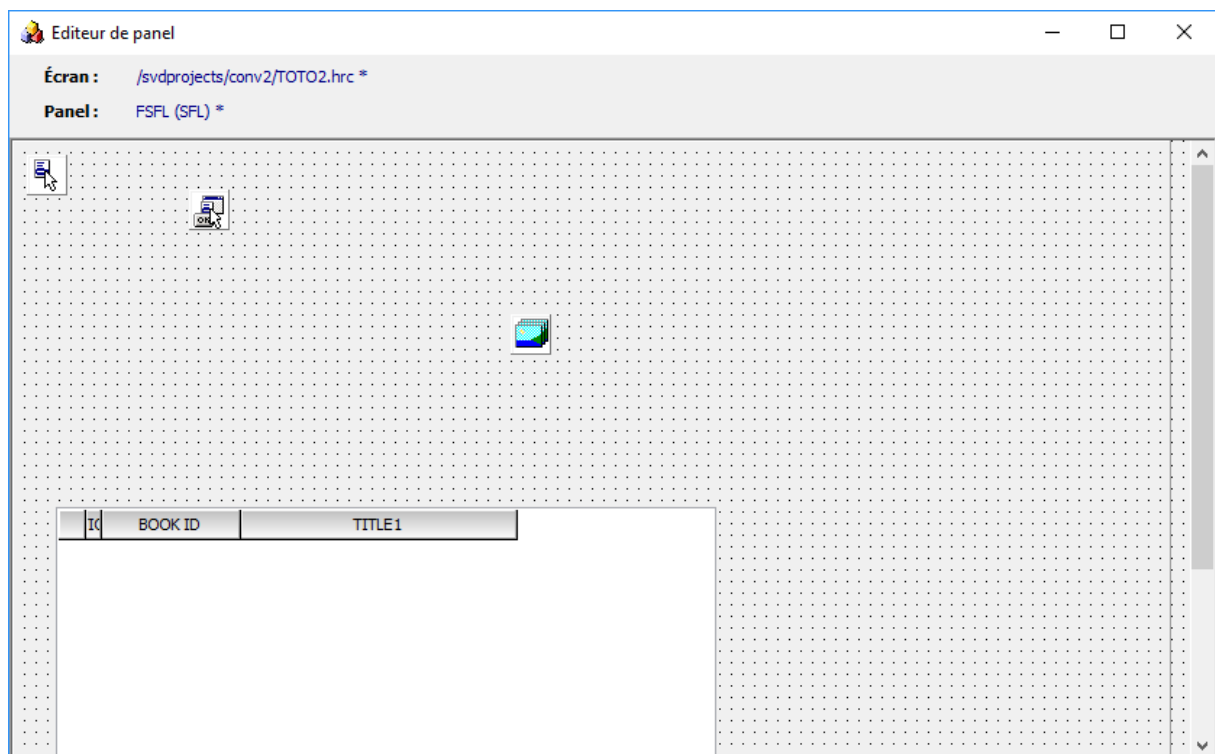
Le code est identique à celui du double click, seule la valeur de l'option est modifiée.

**Remarque** : Si vous avez coché une des options “imprimer” ou “exporter” dans la fenêtre de propriété de la conversion, un menu contextuel a été créé. Ajoutez vos éléments de menu à celui ci.

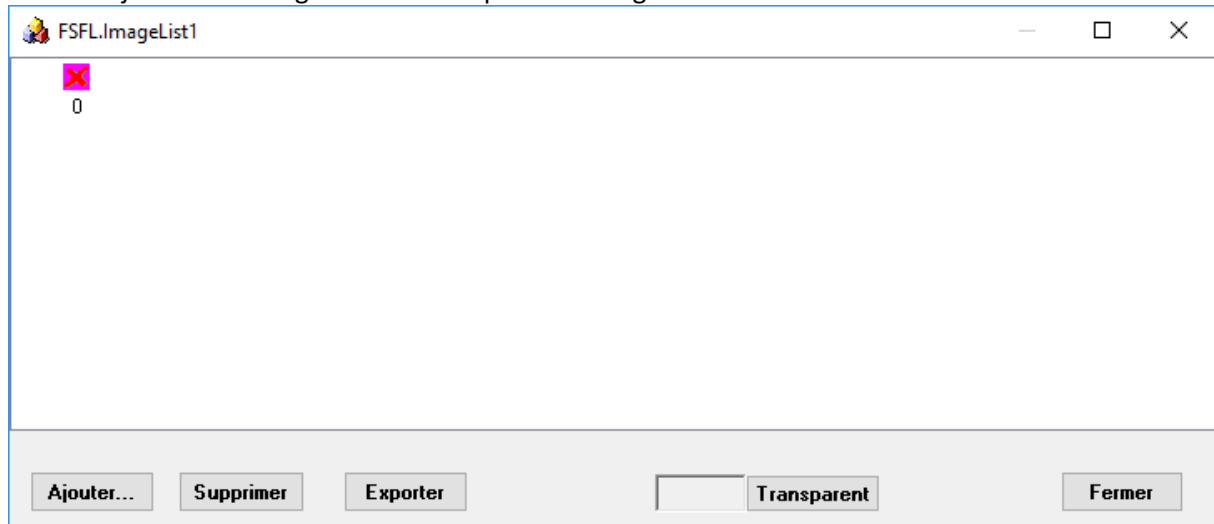
## 8. Ajout d'images dans un menu contextuel

Pour ajouter des images sur les éléments du popup.

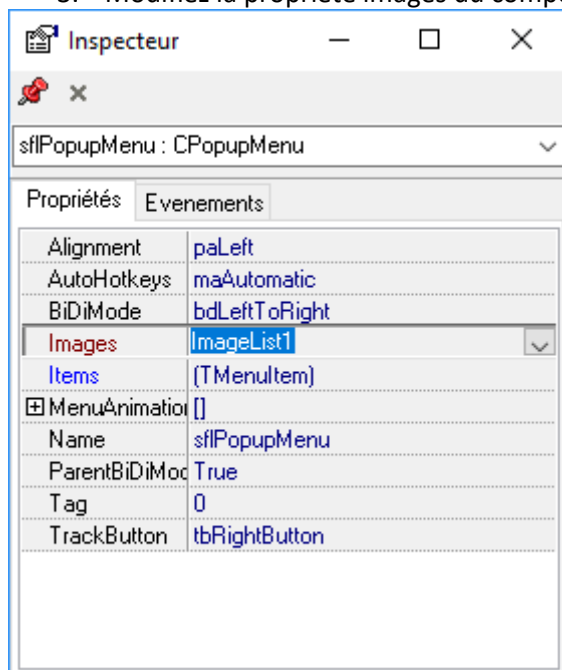
1. Ajoutez un composant CImageList sur le panel



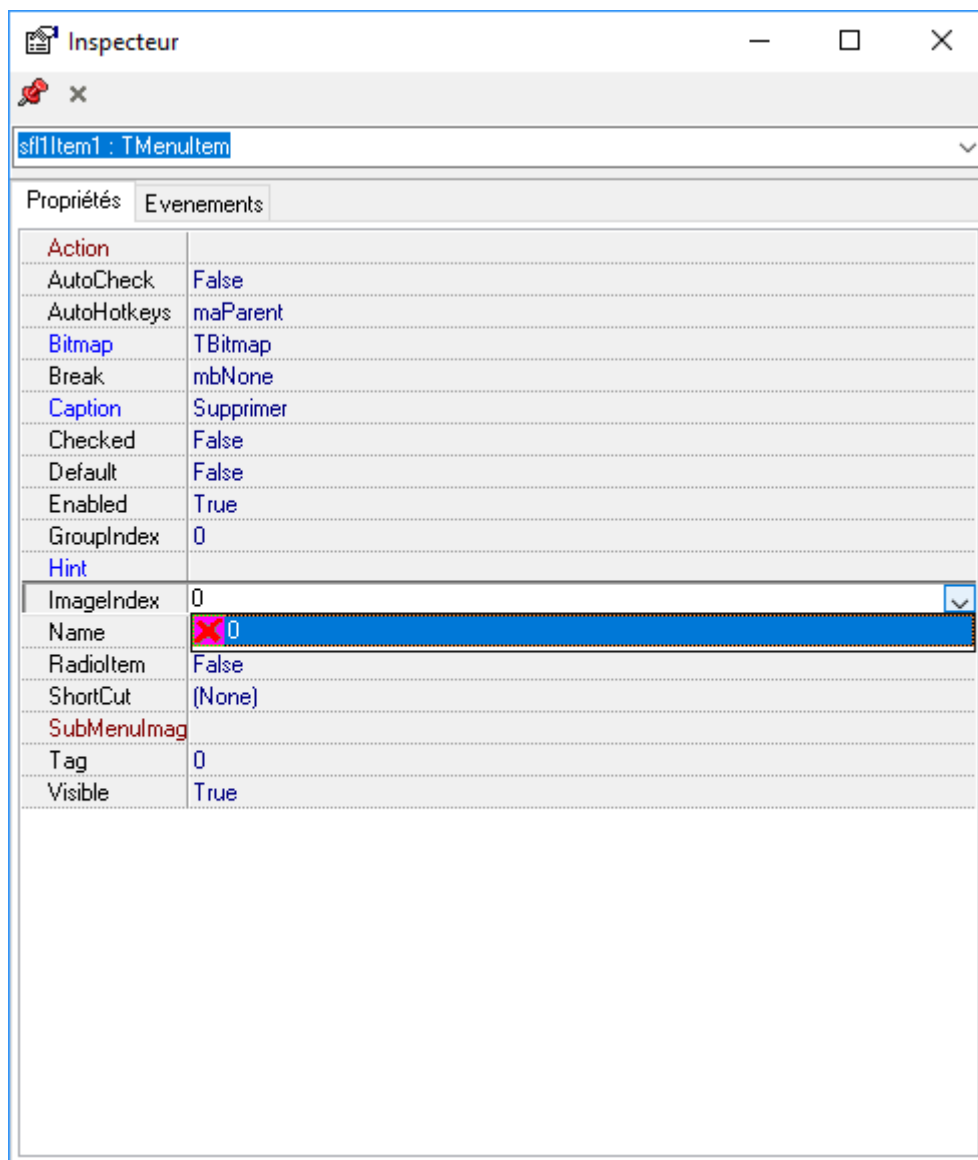
2. Ajoutez des images dans le composant CImageList



3. Modifiez la propriété images du composant CPopupMenu



4. Modifiez les propriétés imageIndex des éléments de menu



## 9. Opération multi lignes

Si la version 5250 gère une option sur plusieurs lignes, vous pouvez ajouter un élément de menu multi lignes.

Par exemple, si l'option 4 permet de supprimer plusieurs lignes, modifiez la propriété options du composant CSFL, ajoutez l'option goMultiSelect :

Name	sf1
Options	{goFixedVertLine,goFixedHorzLine,goVertLine,goHorzLine,goRowSizing,goColSizing,goColMoving,goIndicator,goStar,goRowSelect,goMultiSelect,goBrightness,goRowMoving}
goFixedVertLine	True
goFixedHorzLine	True
goVertLine	True
goHorzLine	True
goRowSizing	True
goColSizing	True
goColMoving	True
goIndicator	True
goStar	False
goRowSelect	False
goMultiSelect	True
goBrightness	True
goRowMoving	False
OptionsGeneration	{TOptionsGeneration}

Créez un élément de menu comme précédemment, puis saisissez le code suivant dans l'évènement onClick :

```

*/EVENT DS3_FSFL_miDelete_OnClick
* -----*
* Description :
* -----*
D Parameters      ds              based(pevtfinf)
D Win              5u 0
D Evt              48
D sf1              s              *
D i                s              10u 0
sf1 = sdGetsf1(stateInfo.F1:'Ds3.format1.sf11':Selected_rows);
if sf1 <> *NULL;
  for i = 1 to sdGetSf1Lines(sf1);
    sdSetCell(stateInfo.F1:'Ds3.format1.sf11':
      'option':sdGetSf1RealRow(sf1:i):'4');
  endfor;
  sdFreeSf1(sf1);
endif;

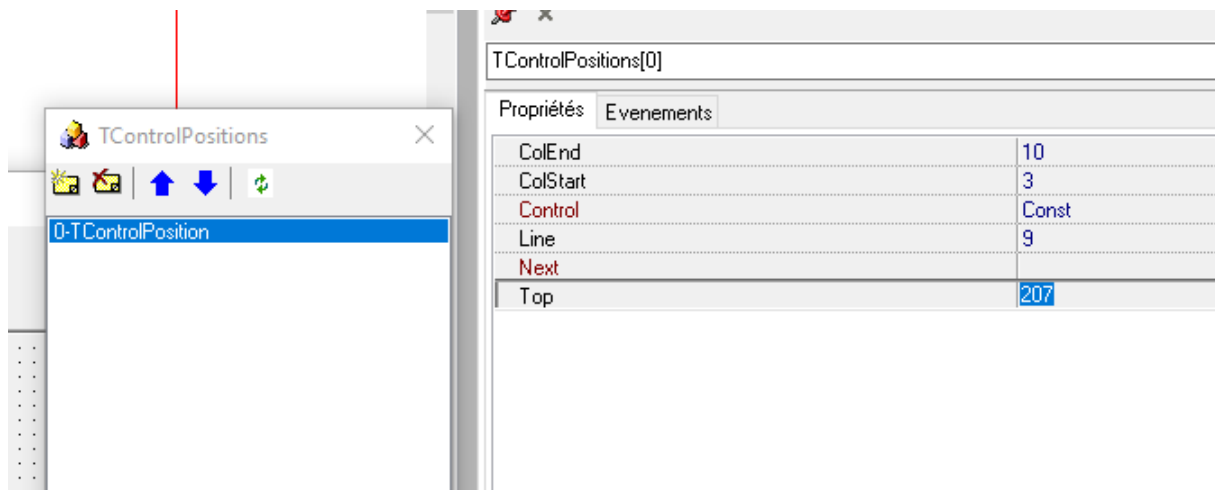
```

## 10. OVERLAY,CLRL

La gestion des mots clefs overlay ou clrl ne se fait pas en fonction de la disposition des composants, mais en fonction de la disposition qu'ils avaient dans l'écran 5250. Les informations sont stockées dans la propriété controlPositions du panel.

Si vous ajoutez un composant (une image ou un bouton par exemple) et que vous souhaitez que ce composant se comporte comme les autres du point de vue des mots clefs overlay er clrl, c'est-à-dire qu'il soient effacés si un format les chevauche, il faut ajouter un élément dans la propriété controlPositions du panel.





### 11. Récupération de la valeur d'une zone.

Pour récupérer le contenu d'une zone écran après la lecture d'un format, interrogez la variable correspondante déclarée dans le handler.

Par exemple, pour une zone nommée FIELD1 en entrée dans le format FORMAT1, vous trouverez la zone stateInfo.FORMAT1\_IN.FIELD1.

Si vous souhaitez récupérer la valeur dans le composant avant la lecture du format, vous pouvez utiliser une fonction silverdev telle que sdGet.

## C. Modification dans le programme copié

Le programme copié peut être modifié. Il est possible par exemple de remplacer un appel à un programme système par un programme silverdev.

## D. Modification dans le handler et le programme copié : USERAREA

### 1. Partage de données

Afin de faire communiquer le programme handler et le programme converti, il est possible d'ajouter à l'instruction workstn handler('HDL0001') un second paramètre :

workstn handler('HDL0001':USERAREA)

ou userarea est une dataStructure de votre choix.

Le programme handler a accès à cette structure de données via la variable handlerParm.userArea

Exemple

Source : LIB2/H/USERDATA

```
D dataPgm0002...
D                                ds          template qualified
D stuff1                        30          varying
D stuff2                        10u 0
```

Source du programme copié :

```
/copy h,USERDATA
D userData          ds          likeDs (dataPgm0002)

FSCR0002 Cf e          workstn
handler('HDL0002':userData)

Avant l'instruction write format0 :

userData.stuff1 = 'hello world';
```

Dans l'écran silverdev, sur le format FORMAT0, on ajoute un label nommé label1.

Source du programme handler :

```
/copy h,userData

D userData          ds          likeDs (dataPgm0002)
D                                based(ptrUserData)

//Dans writeformat, avant l'instruction

//sdhWriteFmt(F1:'FORMAT0')

ptrUserData = handlerParm.userArea;
sdSetString(F1:'DS3.FORMAT0.Label1':'caption':
            userData.stuff1);
```

La variable handlerParm est un paramètre d'entrée ajoutée automatiquement au programme généré.

Le programme copié gère la logique du programme et le handler gère l'écran, mais n'a pas connaissance des données dans le programme copié.

L'utilisation du paramètre userarea permet de palier à cela.

## 2. Passage de pointeurs de procédures

Afin de garder le principe de séparations des rôles.

Le programme d'origine accédant aux données, et le handler proposant l'interface, nous allons voir comment ajouter un graphique dans le handler et lire les données dans le programme d'origine.

#### **a) Programme d'origine**

Dans le programme d'origine, on ajoute la data structure suivante :

```
D UserArea      ds
D
D ptrRefreshGraphic...
D               *   procptr
D
inz(%paddr(refreshGrahpic))
```

La déclaration du fichier écran se fait ainsi :

```
FSCR0002  Cf   e           workstn
handler('SCR0002':userarea)
```

La fonction refreshGraphic qui fera la lecture des données et le remplissage du graphique est la suivante :

```

P refreshGrahpic...
P                                     B
D                                     pi
D win                               5u 0 value
D component                         50   varying value

D date1                             ds           qualified
D day                               2   0
D                                     1   inz('/')
D month                             2   0
D                                     1   inz('/')
D year                              4   0

D date2                             ds           qualified
D day                               2   0
D month                             2   0
D year                              4   0
Dall                                1   8   0

/free
sdSeriesClear(win:component);
setll pData.idBook sddmorder1;
reade pData.idBook sddmorder1;
dow not %eof(sddmorder1);
date1.day = day;
date1.month = month;
date1.year = year;
date2.day = day;
date2.month = month;
date2.year = year;

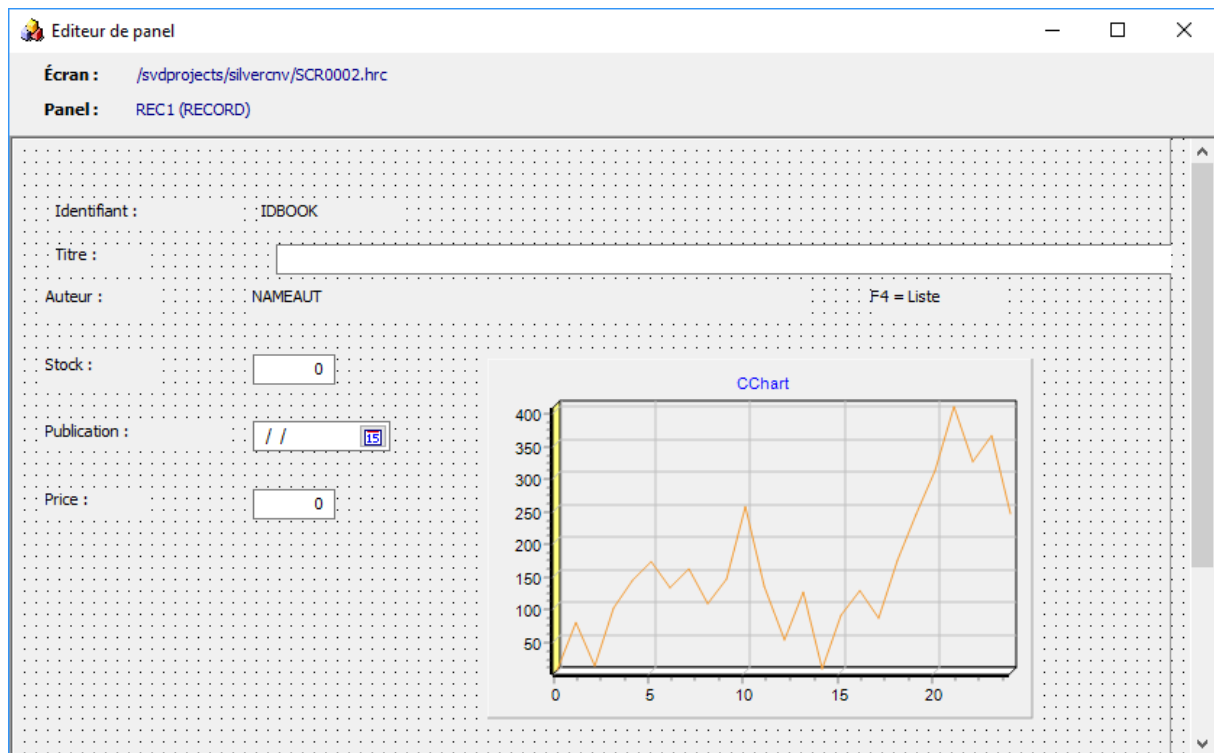
sdAddXY(win:component:date2.all:quantity:date1);
reade pData.idBook sddmorder1;
enddo;

/end-free
P refreshGrahpic...
P                                     E

```

## b) Handler

Dans le handler, on ajoute sur un format un composant CChart:



Dans le source rpg du handler, on ajoute la déclaration de la ds userArea :

```

D UserArea          ds
D
D                  qualified
D                  based(ptrUserArea)
D ptrRefreshGraphic...
D                  *   procptr

```

Puis la définition de la fonction

```

D refreshGraphic...
D                  pr
extProc (UserArea.ptrRefreshGraphic)
D win              5u 0 value
D component        50   varying value

```

Enfin, dans la fonction writeFormat\_Rec1, on ajoute un appel à la fonction refreshGraphic.

```

if handlerParm.userArea <> *null;
  ptrUserArea = handlerParm.userArea ;
  refreshGraphic(stateInfo.F1:'DS3.REC1.Series1');
endif;

```

## VII. Régénération

### A. Introduction

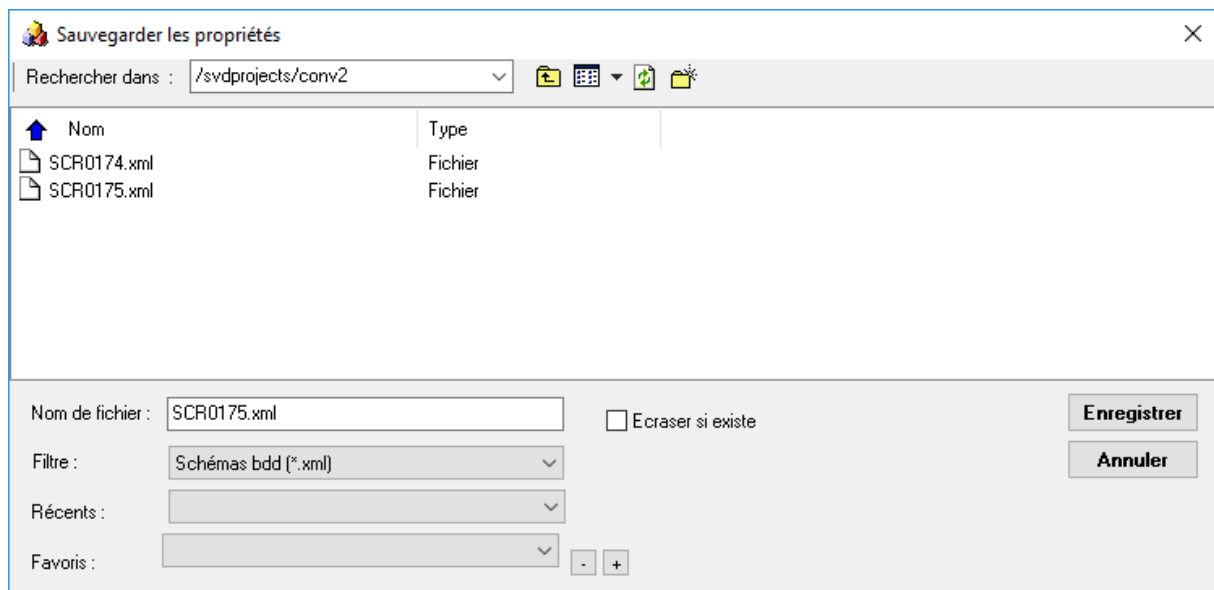
Si l'écran est modifié, vous devez re générer le handler.

Afin de ne pas perdre les propriétés sélectionnées dans la fenêtre de propriétés ou les modifications faites à la main après la conversion, un fichier xml est créé dans lequel toutes ces informations sont enregistrées.

## B. Sauvegarder les propriétés

Une fenêtre de propriétés permet de modifier la largeur de la fenêtre générée, les champs utilisés, les types de composants, les touches de fonction utilisées, la touche de fonctions activée sur la croix de fermeture, les zones dates etc..

Au moment de la génération, une boîte de dialogue propose de sauvegarder ces propriétés.



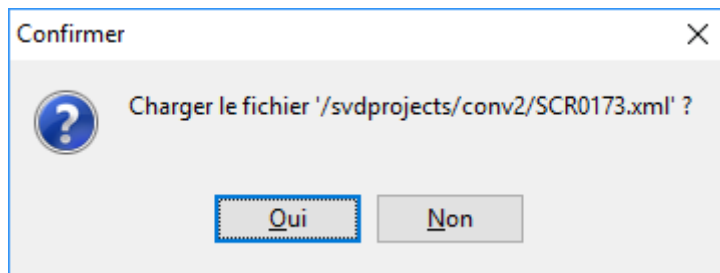
Les propriétés sont sauvegardées au format XML. Il est alors possible de recharger ces propriétés si vous souhaitez re générer le handler.

Le répertoire proposé est le répertoire des sources écrans pour le contexte de destination choisi.

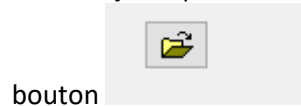
Ainsi, si vous avez oublié de modifier une propriété lors de la génération du handler, vous pouvez relancer la génération en retrouvant les propriétés que vous aviez sélectionné la première fois.

## C. Rechargement des propriétés

A l'ouverture de la fenêtre de propriétés, si un fichier xml du nom de l'écran est trouvé, le designer propose de charger ce fichier :

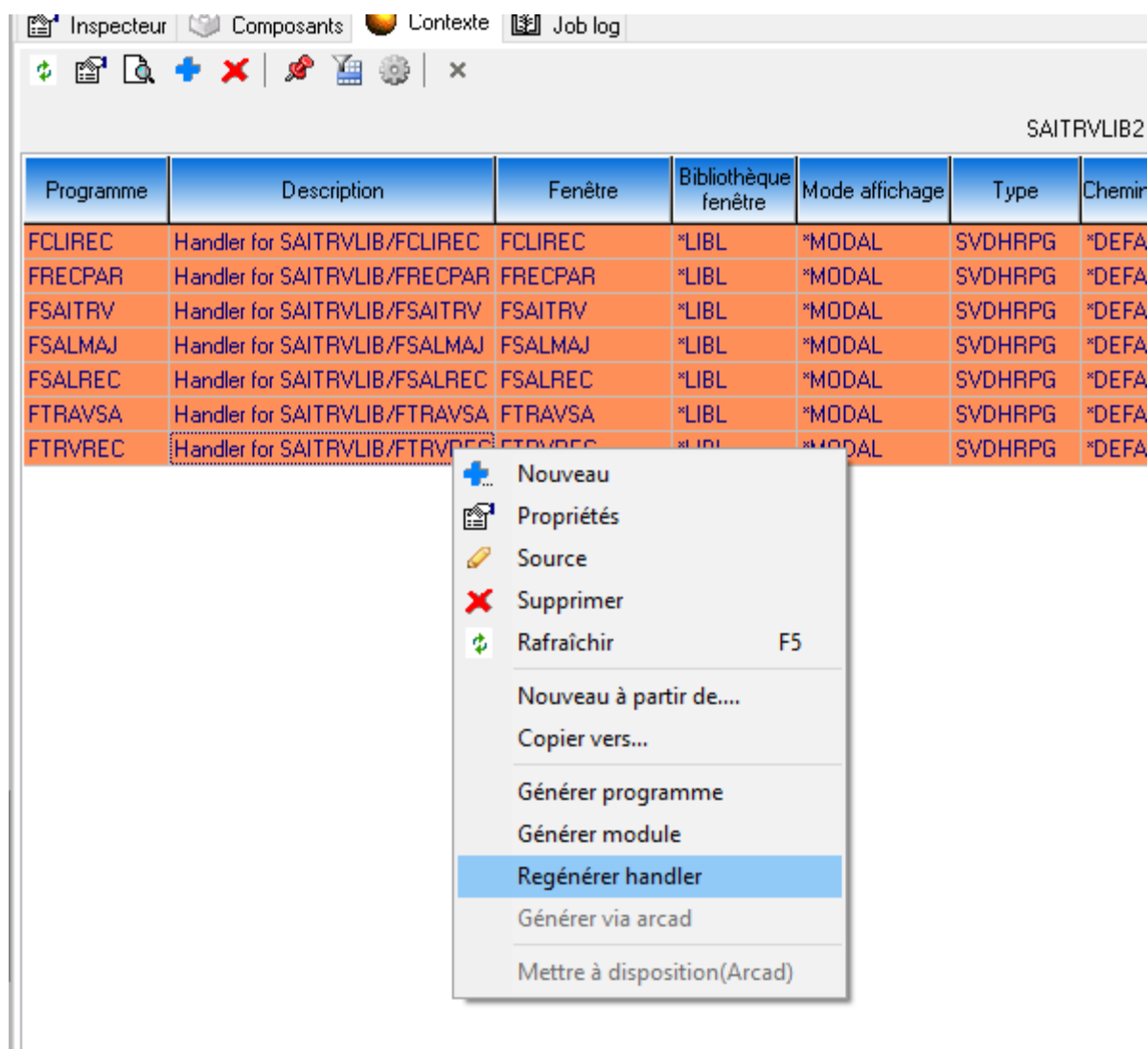


Il est toujours possible de le charger plus tard , ou de choisir un autre fichier xml en utilisant le



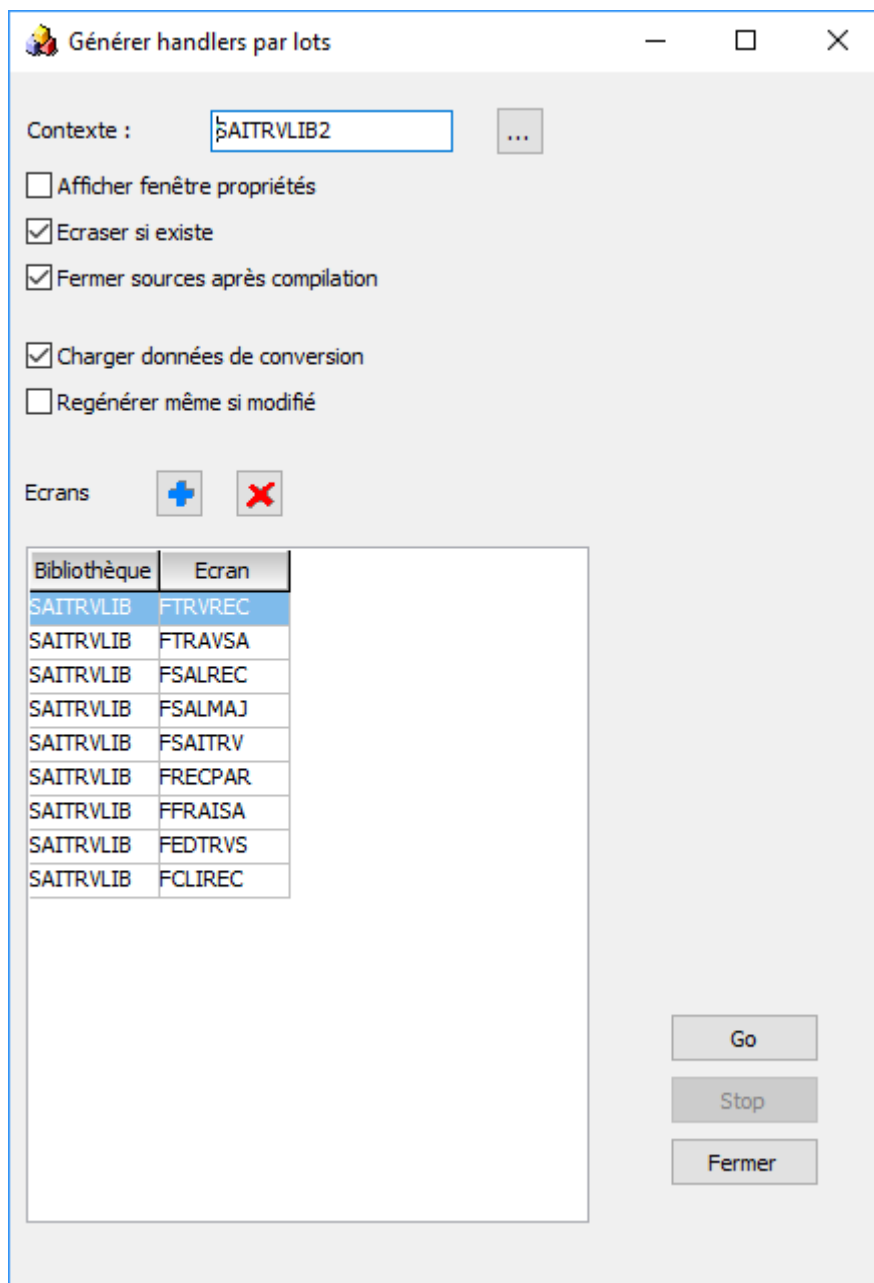
## D. Menu Régénérer handler

Afin de faciliter la re génération des handlers, vous pouvez effectuer un click droit dans le contexte, et sélectionner "Régénérer".



Designer supprime les handlers sélectionnés et affiche la fenêtre de génération par lot





Cochez la case « Charger données de conversion » pour reprendre les propriétés dans la fenêtre de propriété.

## E. Sauvegarde des modifications manuelles

Si vous avez modifié manuellement le handler (écran ou code rpg), vous devez mettre à jour ce fichier en utilisant le menu suivant :

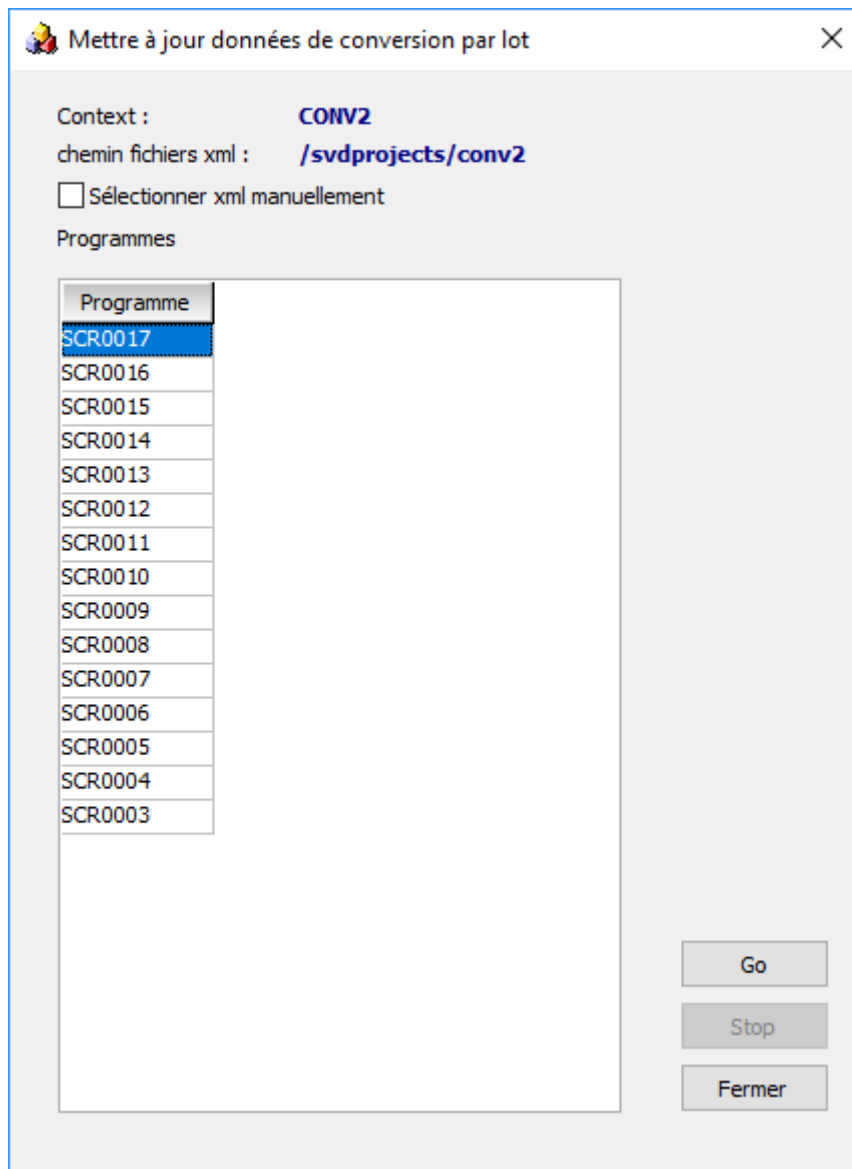
SCR0007	Handler pour CONV1/SCR0007	*PGM	*LIBL
SCR0007B	Handler for CONV1/SCR0007B	SCR0007B	*LIBL
SCR0008	Handler for CONV1/SCR0008	SCR0008	*LIBL
SCR0009			*LIBL
SCR0009B			*LIBL
SCR0010			*LIBL
SCR0011			*LIBL
SCR0012			*LIBL
SCR0013		F5	*LIBL
SCR0014			*LIBL
SCR0015			*LIBL
SCR0016			*LIBL
SCR0017			*LIBL
SCR0018			*LIBL
SCR0019			*LIBL
SCR0020			*LIBL
SCR0021			*LIBL
SCR0022			*LIBL
SCR0024			*LIBL
SCR0025			*LIBL
SCR0025B	Handler for CONV1/SCR0025B	SCR0025B	*LIBL

Remarque : Pour cette opération, designer a besoin d'ouvrir les sources écran et rpg.

Une boîte de dialogue résume la liste des programmes dont les données de conversions vont être mis à jour.

Si la case à cocher « Sélectionner xml manuellement » est cochée, une boîte de dialogue invite l'utilisateur à sélectionner le fichier xml à mettre à jour.

Si elle n'est pas cochée, le fichier xml portant le même nom que le programme dans le répertoire définie pour le contexte sera mis à jour.



## F. Suppressions de composants

Il peut être plus facile de supprimer des champs dans l'écran généré que de cocher les éléments dans la fenêtre de propriétés.

Il est possible que du code rpg soit lié à ces champs supprimés.

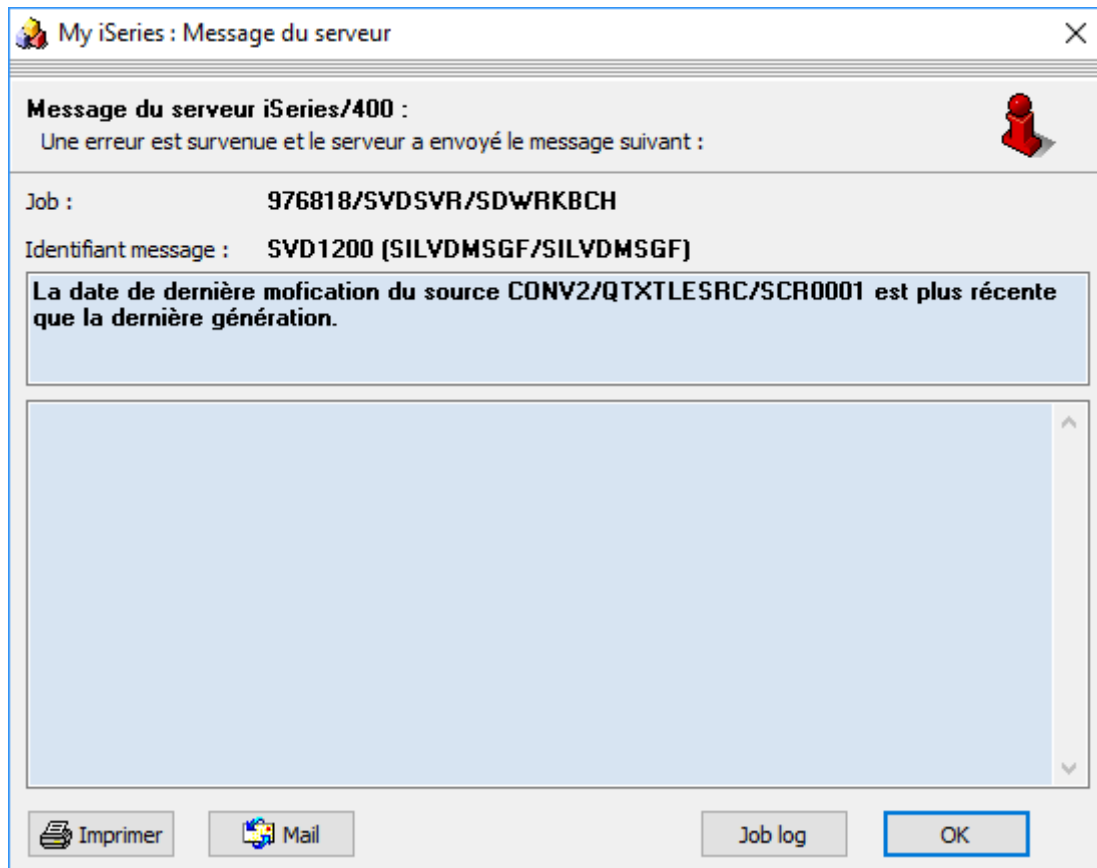
Dans ce cas, il faut Faire une sauvegarde des modifications manuelles, puis une régénération.

La sauvegarde des modifications manuelles va enregistrer le fait que le composant n'existe plus, et la régénération va en tenir compte (le composant ne sera pas généré, le code non plus)

## G. Protection

### 1. Sans charger les données de conversion

Si la case à cocher « charger données de conversion » n'est pas cochée (ou si le fichier n'est pas trouvé), designer s'assure que la date de dernière génération est plus récente que le source de l'écran et du membre dans qtxtlesrc.



### 2. En chargeant les données de conversion

Si la case à cocher « charger données de conversion » est cochée, designer s'assure que la date du fichier de données de conversion au format xml est plus récent que le source de l'écran et du membre dans qtxtlesrc.

Dans ce cas, il faut mettre à jour le fichier des données de conversion. (Voir chapitre « Sauvegarde des modifications manuelles »)

### 3. Forcer la protection

Pour forcer la regeneration, cochez la case "Regénérer même si modifié"

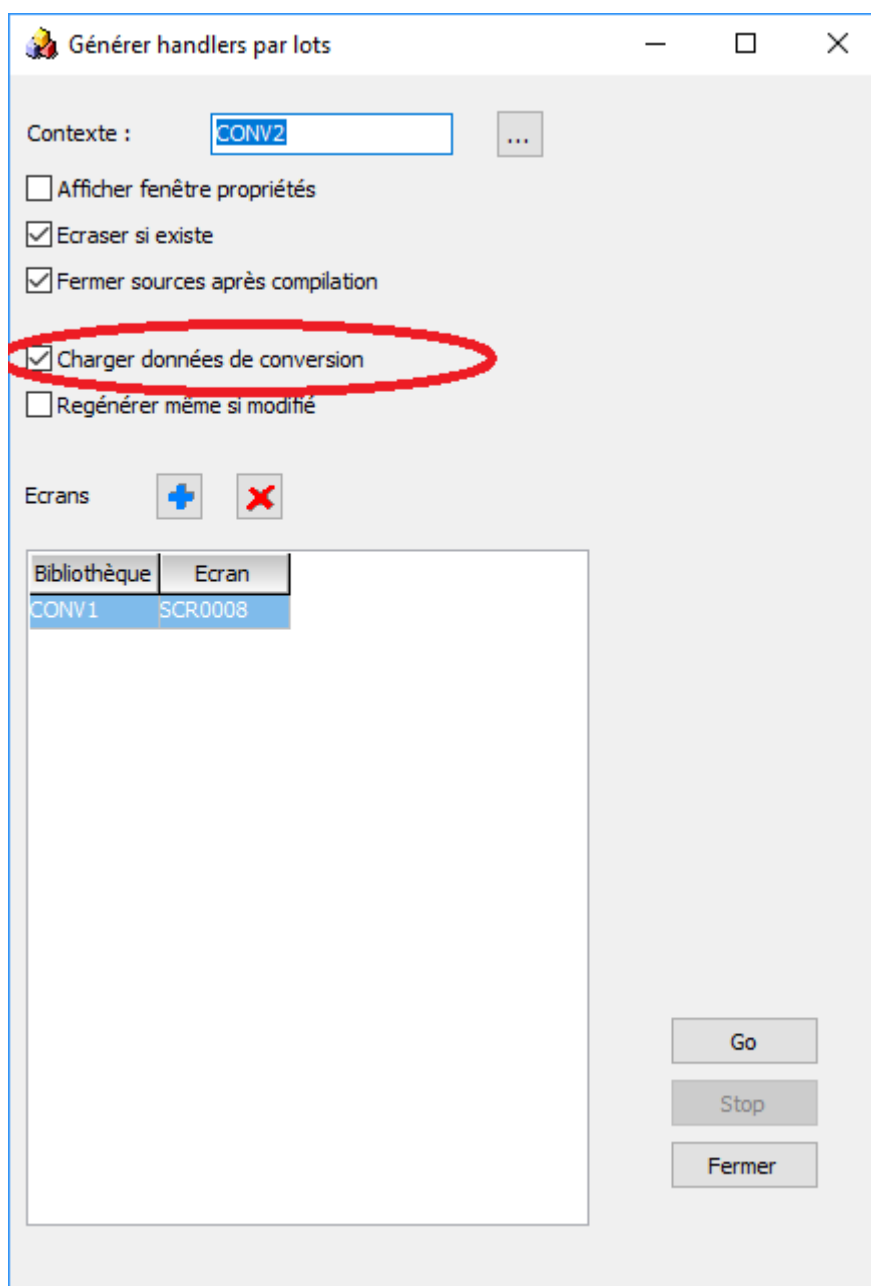
## VIII. Gérer les évolutions de la version 5250

### A. Gérer les évolutions de l'écran d'origine

Si l'écran d'origine est modifié, il faut re générer le handler, voir le chapitre précédent.

#### 1. Rechargement des modifications manuelles.

Vous pouvez ensuite lancer une re génération en demandant à charger ce fichier.



Les modifications manuelles sont alors ré appliquées

## B. Gérer les évolutions du programme d'origine

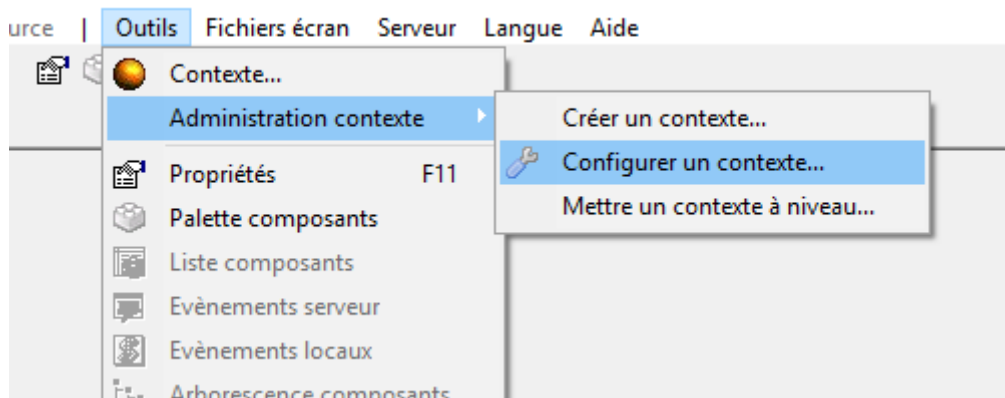
Si le programme d'origine évolue, il faut reporter les modifications dans le programme copié.  
Une autre possibilité est de ne pas copier le programme d'origine, mais d'ajouter des instructions de pré-compilation. Les modifications de type métier seront donc effectives pour les deux versions.


Il est aussi possible de faire évoluer la version copiée sans faire évoluer le programme d'origine.

## IX. Paramétrage des valeurs par défaut

Il est possible de configurer les valeurs par défaut proposées dans la fenêtre de propriétés.

Procédez comme suit :



 Configuration du contexte

Général
 Reprise

**Valeurs par défaut pour la génération de handlers**

Largeur fenêtre

Hauteur fenêtre

Marge

Touche de fonction onClose
 

F3

F12

Conversions
 


Champs 6,0


Champs 7,0

Champs 8,0

SFL
 

☒ Ajouter impression SFL
 ☒ Ajouter export SFL
 ☒ Alternier couleurs lignes SFL
 Lignes impaires :
 
 Lignes paires :
 
 Hauteur entête
 
☒ Colonne de gauche

 OK

 Annuler

## A. Touche de fonction onClose

Pour la touche de fonction activée sur la touche de fonction onClose, la première valeur sera cherchée en priorité, puis la seconde, puis la troisième.

## B. Conversions

Ce paramétrage permet de sélectionner automatiquement un type de conversion pour les champs numériques.

## C. SFL

Ce paramétrage permet de donner des valeurs par défaut sur les composants de type CSFL créés.

# X. Contraintes

Du fait de l'utilisation de la technologie RPGOA, le système d'exploitation minimum est V6R1. Une version V7R1 est mieux car elle permet d'utiliser le modèle de stockage Teraspace. Certains handlers ne compileront pas sans cette option.

Seuls les programmes RPG peuvent être convertis.

La conversion nécessite d'avoir les sources du programme RPG

Le programme d'origine ne doit pas faire appel à des interfaces 5250.

- Tous les programmes appelés doivent être eux aussi convertis.
- Il ne peut y avoir d'appel à des programmes système, tels que wrksplf
- ligne de commande
- Le programme rpg d'origine ne doit pas générer de flux 5250.

Seuls les écrans définis en externe sont gérés.

Lors de la compilation du programme copié et de son exécution, il faut que l'écran 5250 soit en ligne.

Lorsqu'un programme ne satisfait pas à contraintes citées plus haut.

Il est possible de modifier le programme copié, ou de réécrire entièrement le programme en silverdev classique grâce à la compatibilité des programmes silverdev classique et silverdev convertis.



## XI. Programmes RPG 3

L'instruction handler n'est disponible que en RPGLE. Pour convertir un programme RPG ver RPGLE, utilisez la commande CVTRPGSRC

Remarque :

Si la commande CVTRPGSRC aboutit au message suivant :

RNS9378 Fichier de consignment QRNCVTLG introuvable dans la bibliothèque \*LIBL.

Vous pouvez créer ce fichier avec la commande crtdupobj (et non cpyf comme mentionné dans l'aide du message) en copiant le fichier qrpgle/qarncvtlg

## XII. Solutions de remplacement

### A. Appel d'un programme système

Les programmes systèmes peuvent être réécrits en silverdev à l'aide notamment des apis IBM. Ces programmes pouvant être appelés dans plusieurs programmes, ce travail pourra être réutilisé. Experia fournit déjà des programmes de ce type, et le nombre de programmes systèmes fournis par Experia va augmenter avec le temps

### B. Ligne de commande

Si une ligne de commande est affichée dans le programme d'origine, il est possible d'ajouter un composant CCmdDialog.

Exemple :

### C. Dsply

Un appel à dsply, peut être remplacé par la fonction sdShowMessage.

Il faut ajouter les instructions suivantes au programme :

```
H bnddir('SILVERDEV')  
H DFTACTGRP(*NO)  
/copy h,silverdev
```

### D. Suppression

Si un programme ne peut pas être ré écrit car il est trop lié au 5250, ou si vous souhaitez reporter son écriture dans le temps, vous pouvez désactiver la touche de fonction qui appelle ce programme. (Voir chapitre Désactivation de touches de fonctions)

## E. RECVF

Chez un client, on a trouvé le code suivant :

DCLF	FILE (WD001FM) RCDFMT (WD001WW)
SNDF	RCDFMT (WD001WW)
RCVF	RCDFMT (WD001WW) WAIT (*NO)
DLYJOB	DLY (3)

Afin que la fenêtre se ferme au bout de 3 secondes.

Nous avons remplacé par ceci :

H DFTACTGRP(*NO)	
FWD001FM CF E	WORKSTN
F	handler('WD001FM')
/free	
*inlr = *on;	
exfmt WD001WW;	
/end-free	

Sur l'écran silverdev, nous avons ajouté un composant timer réglé à trois secondes, et dans l'évènement ontimer :

*/EVENT DS3_WD001WW_Timer1_OnTimer	
* -----	
--*	
* Description :	
* -----	
--*	
D Parameters	ds based(pevtinf)
D Win	5u 0
D Evt	48
/free	
sdSetBool(stateInfo.F1:'DS3.WD001WW.Timer1':'enabled':*off);	
sdClose(stateInfo.F2);	
/end-free	

## XIII. Champs référence

Si une zone d'un écran est marquée avec le mot clef REFFLD, la bibliothèque du fichier utilisé au moment de la compilation du DSPF est indiquée en dur dans le DSPF.

L'outil de conversion cherchera d'abord le fichier dans cette bibliothèque, puis s'il ne le trouve pas, dans la liste de bibliothèque.

## XIV. Licences

Pour utiliser l'outil de conversion, il faut avoir une licence développement comprenant l'option conversion.

Si l'option conversion n'est pas active ou la date limite est dépassée, le menu "Fichier Ecran/Générer un handler" est désactivé.

Pour exécuter les programmes convertis, la licence runtime de silverdev suffit.

Donc, même lorsque la licence conversion est passée, les programmes convertis fonctionnent.

Il est toujours possible de modifier à la main les programmes convertis lorsque la licence conversion est passée.

## XV. Plusieurs écrans dans un programme

si un programme fait appel à plusieurs écrans, il faut créer le handler pour chacun des écrans.

Exemple :

```
H DFTACTGRP(*NO)
FSCR0167  Cf  e      workstn handler('HDL0167')
FSCR0168  Cf  e      workstn handler('HDL0168')

/free
*inlr = *on;
dow not *in03;
  exfmt RECORD1;
  if *in04;
    exfmt RECORD2;
  endif;
enddo;
/end-free
```

## XVI. Erreurs fréquentes

CPF4103	Le mot clef handler n'est pas ajouté, et le programme cherche l'écran d'origine. Il est aussi possible que le programme copié n'ait pas été compilé, c'est alors le programme d'origine qui est appelé.
CPF4103	Un programme dans la chaine de programmes n'a pas été converti et le programme d'origine a été appelé.
MCH3401	Le programme handler n'a pas été compilé

SVD4130	Le programme copié a été compilé avec une version différente de l'écran 5250 Solution: recompiler le programme
SVD4131	Le programme handler a été compilé avec une version différente de l'écran 5250 Solution : re-générer et compiler le programme
MCH3402	L'écran 5250 a été supprimé pendant que le programme s'exécutait. Le problème survient lors de la fermeture de la fenêtre.
User	Si dans les programmes rpg 5250 vous utilisez le champ à l'offset 254 de la sds, en silverdev, ce champ vous donnera le profil qui a démarré les serveurs silverdev. Il faut remplacer par le champ à l'offset 358 (à 367)

## XVII.Exemple complet

### A. Présentation

Dans la bibliothèque Silverdemo, se trouve des programmes 5250.

Les programmes sont PGM0001 à PGM0003.

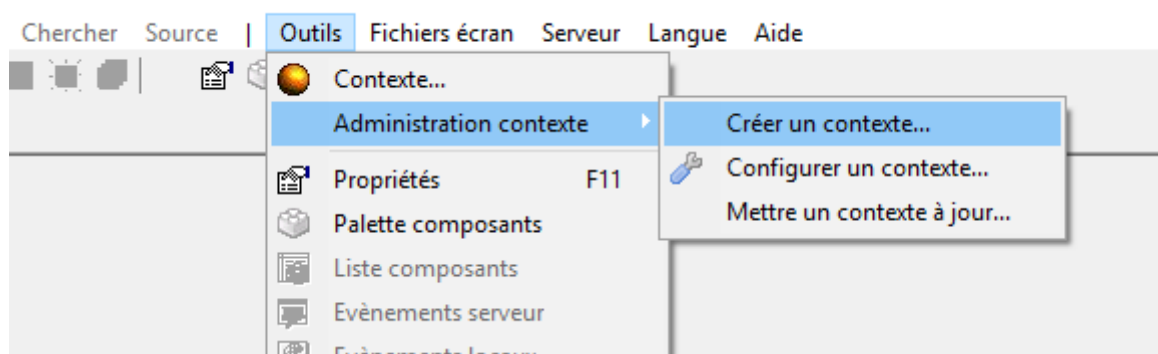
Les écrans respectifs sont SCR0001 à SCR0003.

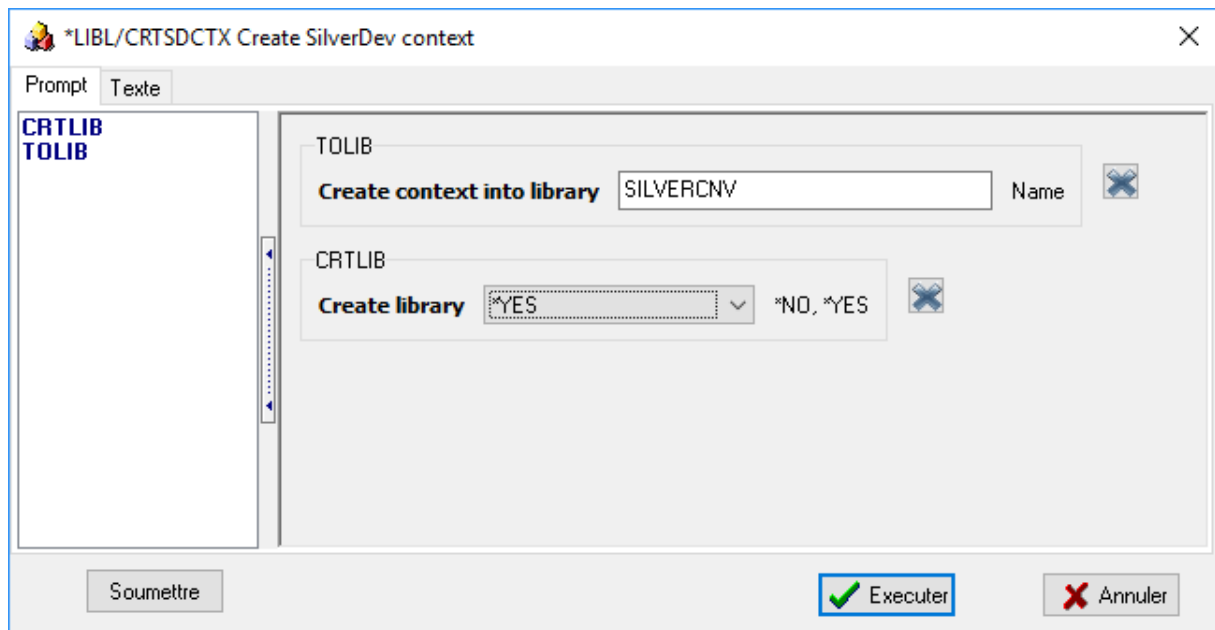
Le premier écran affiche une liste de livre, l'option 2 permet d'accéder à la fiche livre, et la touche F4 sur la fiche livre permet de sélectionner un auteur.

Les programmes originaux, les fichiers de base de données et les écrans 5250 sont dans la bibliothèque SILVERDEMO.

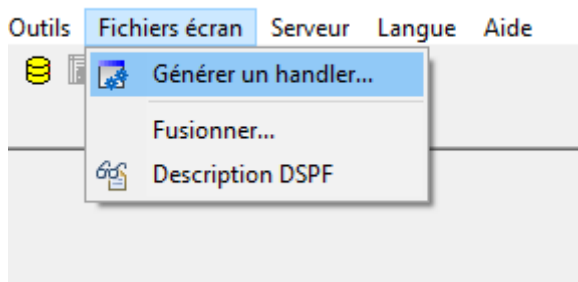
### B. Création du contexte

On commence par créer une bibliothèque SILVERCNV et un contexte dans SILVERCNV.





### C. Génération des handlers



Génération de handler

Écran 5250

Bibliothèque : SILVERDEMO ...

Nom : SCR0001 ...

↓

Programme Silverdev

Contexte : SILVERCNV ...

Programme : SCR0001

Écran : \*PGM

Type : SVDHRPG

Description : Handler pour SILVERDEMO/SCR0001

OK Annuler

## D. Copie des programmes originaux

On copie les programmes originaux dans silvercnv et on ajoute l'instruction handler dans la déclaration des écrans.

```
1 | h dftactgrp(*no)
2 | FSCR0001 Cf e workscr handler('SCR0001')
3 | F afile(fsfl:rrnl)
4 | FSDDMBKS if e k disk
5 |
6 |
7 | D rrnl s like(wdpos)
8 | D count s 2 0
9 |
10 | D load...
11 | D pr
12 |
13 | D startLoad...
14 | D pr
15 |
16 | D saveSearch s like(FldSearch)
17 |
18 |
19 | D pgm2 pr extpgm('PGM0002')
20 | D pData likerec(fbooks)
```

## E. Lancement du programme converti

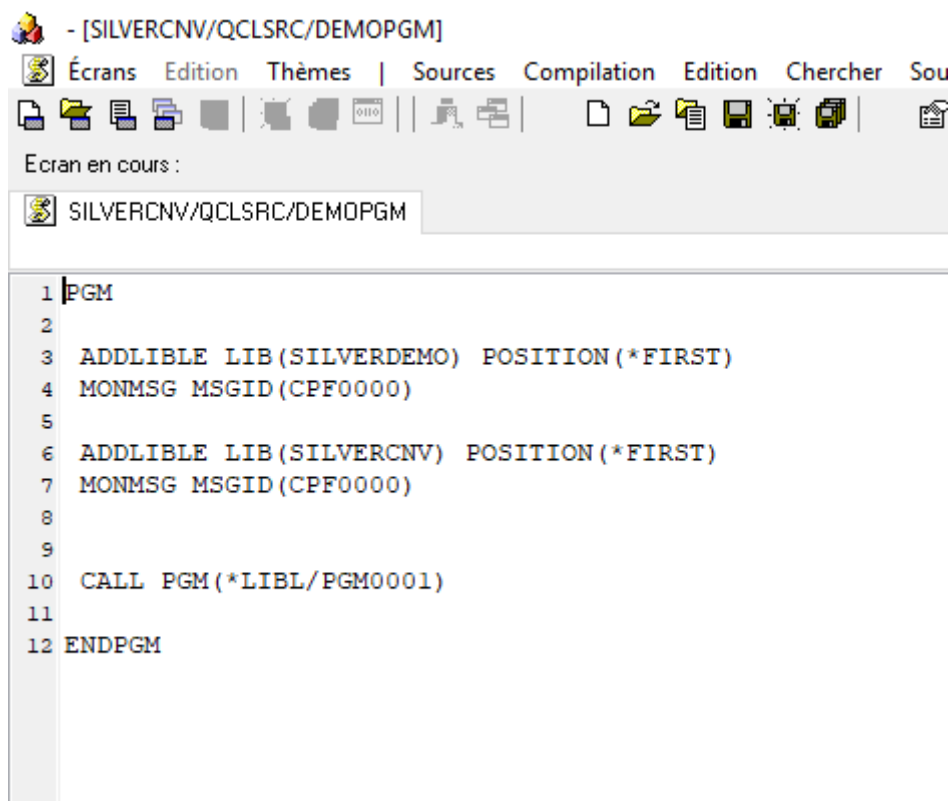
Dans SILVERCNV/QCLSRC, on écrit un programme de lancement.

On ajoute SILVERDEMO et SILVERCNV en liste de bibliothèques.

SILVERCNV doit être avant SILVERDEMO pour que les programmes convertis soient utilisés et non les programmes originaux.

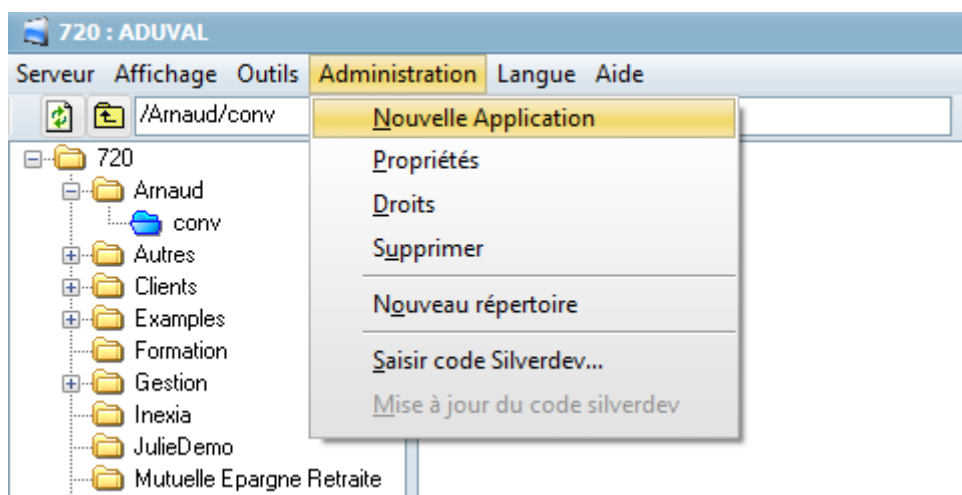
SILVERDEMO doit être en liste de bibliothèque car les écrans 5250 doivent être en ligne.

De plus, les fichiers de base de données sont dans SILVERDEMO.



On ajoute une icône dans MyDesk:

(Le profil utilisé pour la connexion doit être dans le fichier SILVERDEV/PSVDADM pour que le menu administration apparaissent.)





**Modification** [X]

**Titre :**  
Programme de démo

**Nom du fichier app :**  
DemoPgm

**Commande :**  
call silvercnv/demoPgm ...

**Description :**

☐ Execution unique

**Icône :** [Icon] ... [Pencil] [Text Box]

**Icône interne :** [Icon] ...

**Rang :** [ -1 ]

[OK] [Annuler]

Lancement de l'application

Programme de démo


2 = Modifier

Rechercher :

OPTION1	BOOK ID	TITLE1
	1	Brown Bear
	2	From Head to Toe
	3	Today Is Monday
	4	Where's Spot
	5	The very hungry caterpillar
	6	The road
	7	Mystic river
	8	Duma key
	9	The black dahlia

More...

F5 = refresh

— □ ×

Identifiant : 3

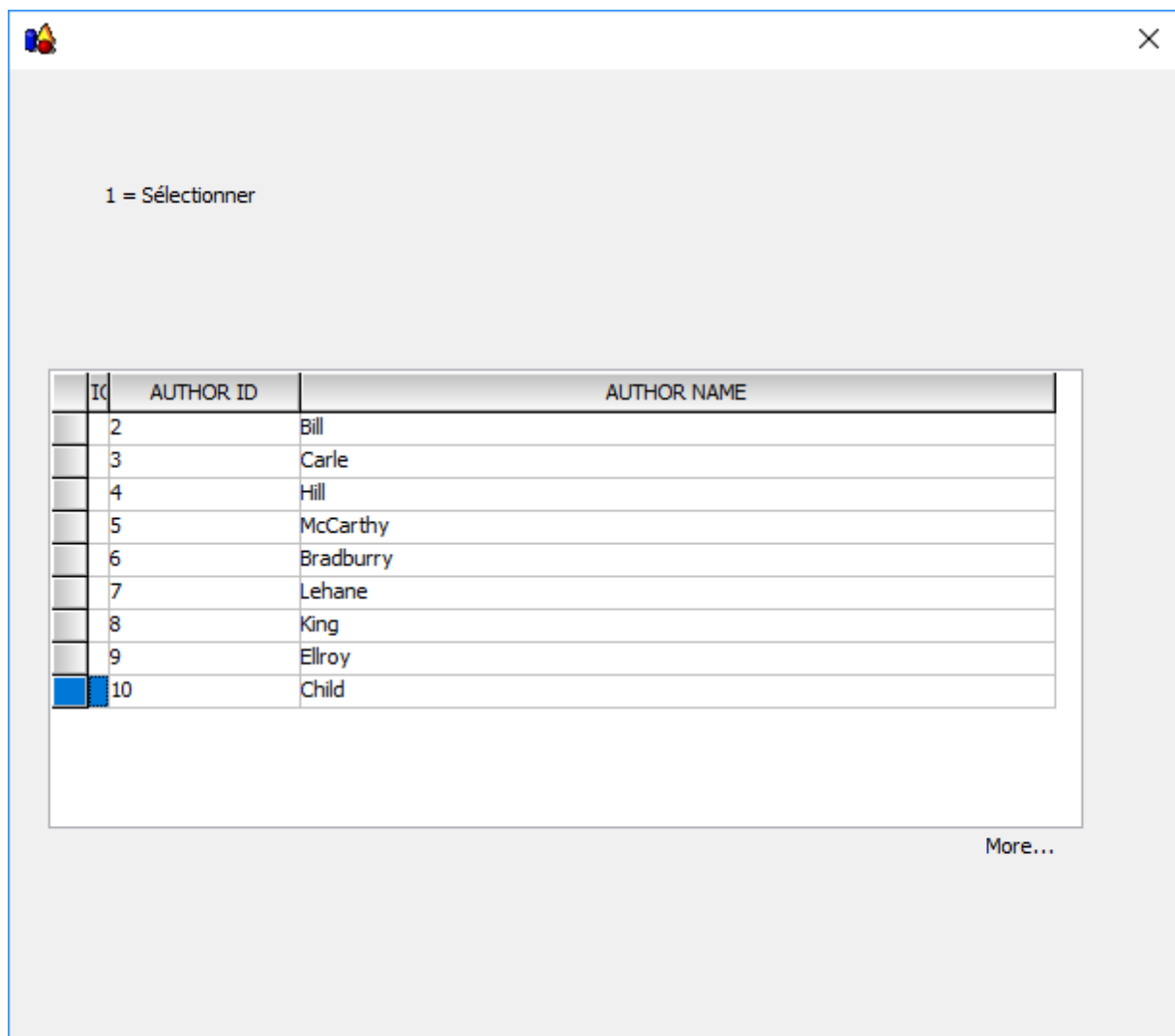
Titre :

Auteur : Carle F4 = Liste

Stock :

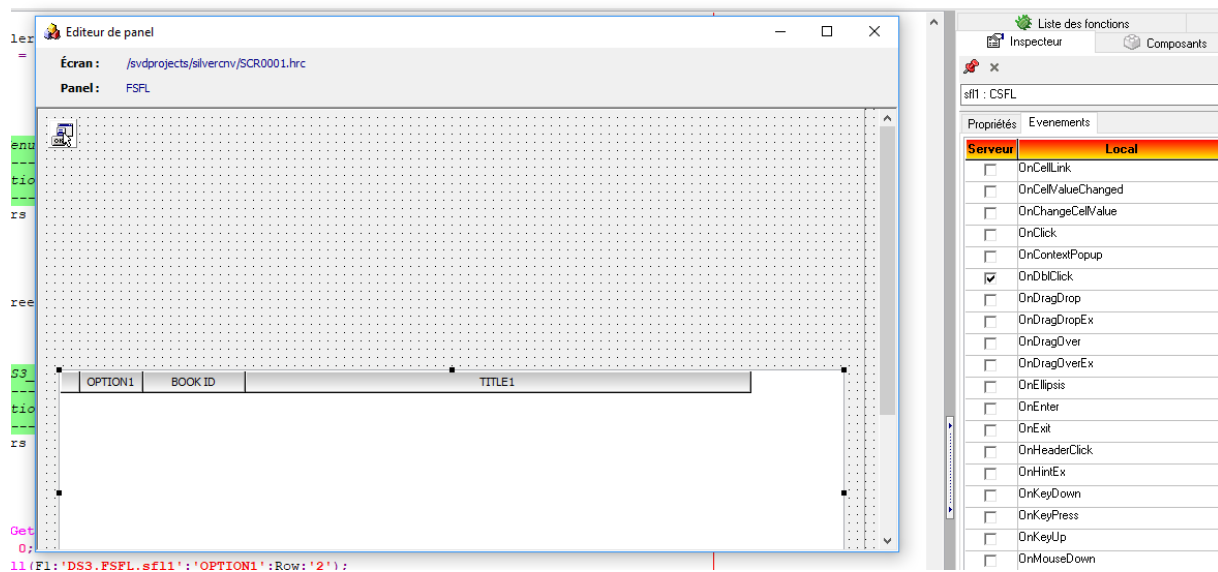
Publication :

Price :



## F. Modifications dans les programmes générés

1. Ajout du double click sur les sous fichiers :



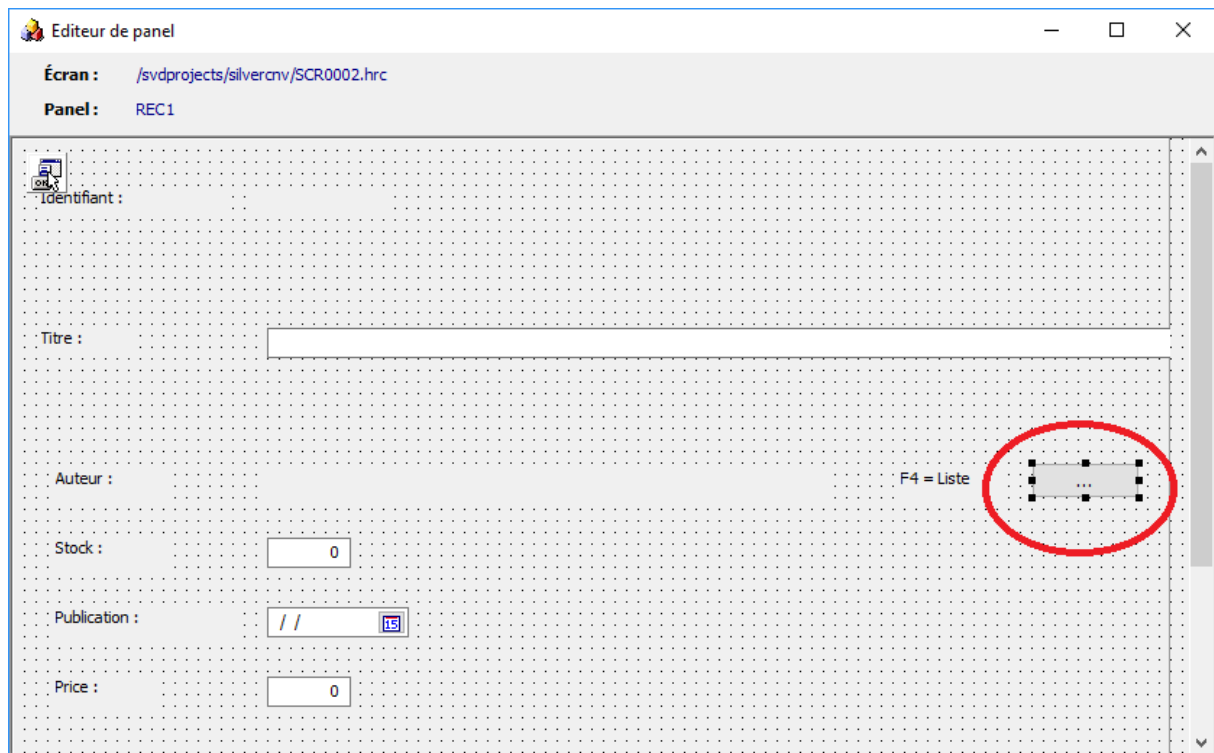
```

*/EVENT DS3_FSFL_sf11_OnDbClick
* -----
* Description :
* -----
D Parameters      ds          based(pevtf)
D Win              5u 0
D Evt              48
D row              s          10i 0
/free
row = sdGetInt(F1: 'DS3_FSFL_sf11': 'rowSelected');
if row > 0;
    sdSetCell(F1: 'DS3_FSFL_sf11': 'OPTION1': Row: '2');
    DS3_FCTLSFL_ENTER_OnExecute();
endif;
/end-free

```

## 2. Ajout d'un bouton dans pgm2

Dans le programme PGM2, on ajoutera un bouton qui ouvrira la même fenêtre que la touche F4.



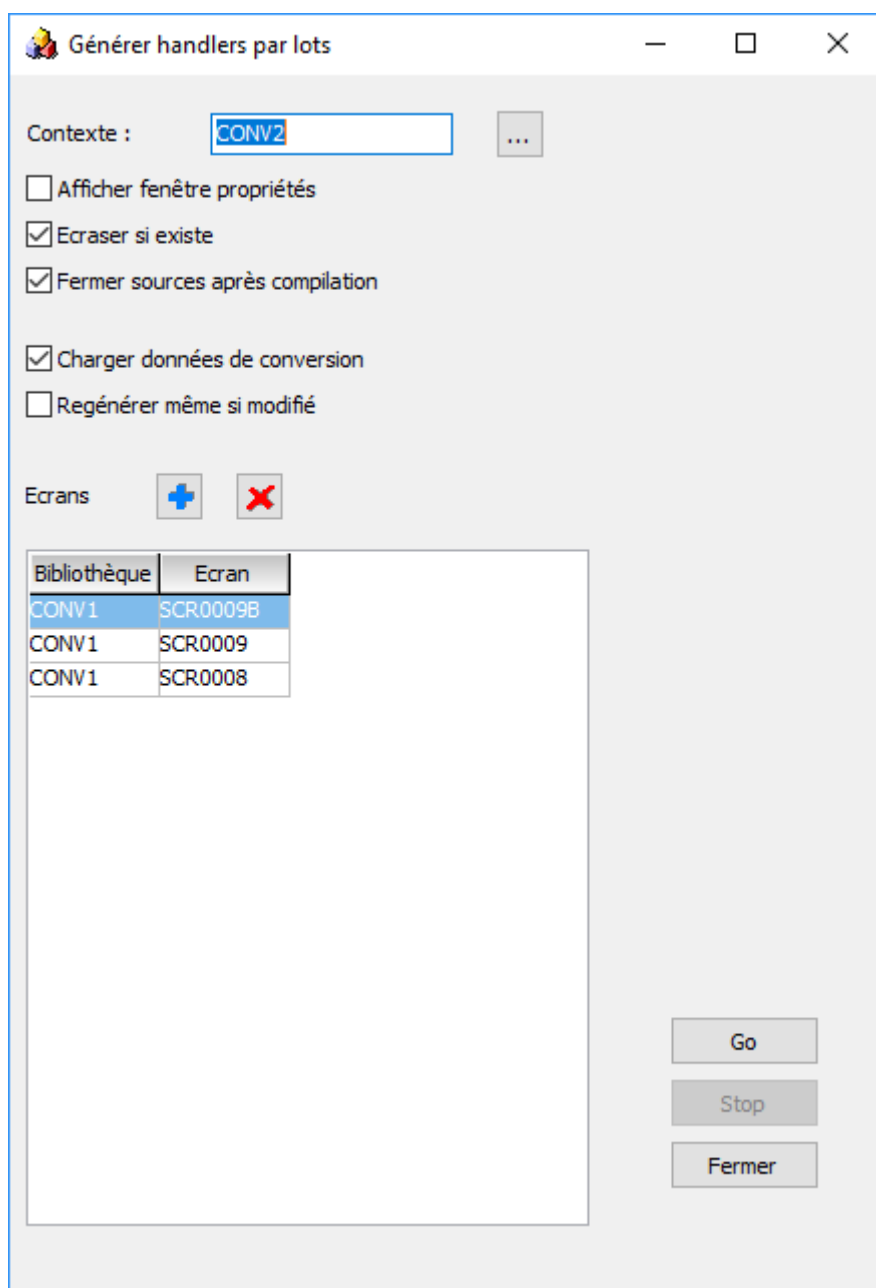
```

*/EVENT DS3_REC1_btnSearchAuthor_OnClick
* -----*
* Description :
* -----*
D Parameters    ds          based(pevtinf)
D Win           5u 0
D Evt           48
/free
  REC1_IN.FLD = 'NAMEAUT';
  DS3_REC1_CF04_OnExecute();
/end-free

```

## XVIII. Génération multiples

Pour générer plusieurs handler, cliquez sur "Fichiers Ecrans/Générer handlers par lots"



## XIX. Mots clefs

Le tableau ci-dessous reprend la liste des mots clefs pour les fichiers DSPF et décrit comment la fonctionnalité liée à ce mot clef est traduite en silverdev :

1	Alarm	sdBeep généré avant sdExfmt. Indicateurs pris en compte.
2	Alias	Alias ne semble pas être supporté par le rpg. Cela n'aurait de toutes façons pas d'incidence sur le handler
		Voir <b>Erreur ! Source du renvoi introuvable.</b>

3	Althelp	Fonction help est appelée
4	Altname	Altname ne semble pas être supporté par le rpg. Cela n'aurait de toutes façons pas d'incidence sur le handler.
5	Altpagedwn	Evènement rollup_onexecute est appelé
6	AltPageUp	Evènement rolldown_onexecute est appelé
7	Alwgph	GDDM non supporté -> mot clef non supporté
8	Alwrol	Non supporté en rpg
9	Assume	Les formats enregistrés dans la variable keepPanels sont affichés à l'ouverture du fichier écran
10	Auto	Equivalant à check
11	Blanks	Test sdGetText = *blanks
12	Blink	Rien de spécial n'est fait
13	Blkfold	Rien à faire car la version silverdev ne fait pas de fold (ascenseurs) Voir : <b>Erreur ! Source du renvoi introuvable.</b>
14	Cann	Ajout d'un composant TAction, shortCut avec la valeur de nn L'écran n'est pas lu
15	Cfnn	Ajout d'un composant TAction, shortCut avec la valeur de nn L'écran est lu
16	Change	Le texte de l'indicateur est affiché dans le designer. L'indicateur de réponse est mis à *on
17	Chcaccel	Caption du radiobutton modifié
18	Chcavail	Pas important
19	Chcctl	Etat du composant modifié. Reste à traiter le message à afficher si zone désactivée et sélectionnée.
20	Chcslt	Rien à faire
21	Chcunavail	Rien à faire
22	Check	CHECK(ME) : test champ non vide CHECK(LC) propriété charcase CHECK(RB) aligné à droite et evalr Check(AB) Si comp, range ou values, ajout de la possibilité d'avoir un blanc CHECK(MF) Test champ rempli CHECK(M10) appel à sdhCheck('M10'...  CHECK(M11) appel à sdhCheck('M11'... CHECK(M10F) onchange, appel à sdhCheck('M10'... CHECK(M11F) onchange, appel à sdhCheck('M11'... CHECK(VN) appel à sdhCheck('VN'... CHECK(VNE) appelle à sdhCheck('VNE'... CHECK(RZ) displayFormat rempli avec des zéros  CHECK(ER) non géré CHECK(FE) non géré CHECK(RL) non géré CHECK(RLTB) non géré
23	Chginpdf	Rien à faire, voir <b>Erreur ! Source du renvoi introuvable.</b>
24	Chkmsgid	Appel à sdRtvMsg
25	Choice	Pour sngchcflid, création d'un radiobutton



26	Chrid	Pas de ccsid sur la partie cliente. Tous les champs sont affichés dans le ccsid du programme rpg
27	Clear	Ajout d'une action
28	Crlr	Des contrôles sont enlevés de la fiche
29	Cmp	Contrôlé à la validation
30	Cntfld	Génération d'un champ normal, prise en compte pour le overlay.  <b>Modifications futures possibles</b>
31	Color	Font.color est modifié. (si green -> clWindowText)
32	Comp	Idem cmp
33	Csrinponly	TabStop est mis à false selon CSRINPONLY
34	Csrloc	Appel à sdGetControlAt
35	Date	%char(%Date) . A faire paramètres du mot clé
36	Datfmt	Pris en compte pour la déclaration des buffers. (obligatoire)
37	Datsep	Pris en compte pour la déclaration des buffers. (obligatoire)
38	Dft	Valeur mise par défaut si champ en output et première écriture
39	Dftval	Valeur mise par défaut si champ en output et première écriture, ou si champ en input
40	Dltchk	Rien à faire, les informations ne sont pas ajoutées dans le fichier écran, donc pas lues par designer  Voir <b>Erreur ! Source du renvoi introuvable.</b>
41	Dltedt	Rien à faire, les informations ne sont pas ajoutées dans le fichier écran, donc pas lues par designer
42	dspatr	Modification de la police, de la couleur, de readonly, de visible et de la position du cursor
	Dspatr(pr)	Readonly. Si dspatr(nd) et dspatr(pr) ou si zone en sortie seule et dspatr(pr) -> zone invisible
	Dspatr(nd)	Si dspatr(pr) -> invisible, sinon, passwordchar = '*'
43	Dspmod	Le panel adéquat est affiché
44	Dsprl	<b>Non géré</b>
45	Dspisz	Pour chaque format, un panel est créé par dspSiz
46	Dup	Non géré
47	Edtcde	%editc généré
48	Edtmsk	Propriété editMask générée en fonction.
49	Edtwrd	Création d'un maskEdit si zéro décimales adaptation de la propriété editmask . Si décimales, création d'un editnum, génération de la propriété displayFormat
50	Entfldatr	Rien à faire
51	Erase	Génération de sdhHideFmt
52	Eraseinp	Mise à blanc des champs déjà affichés, et non protégés. Seulement les mdton si ERASEINP(*DDTON)
53	Errmsg	Erreur affichée à l'exécution. Remarque, l'indicateur de réponse est remis à *off par le système, pas de code à générer pour cela.
54	ErrMsgld	Idem ErrMsg
55	Errsfl	Les messages d'erreurs sont de toute façon toujours mis dans un sous fichier. Les formats internes sont ignorés
56	Fldcsrprg	Propriété next ajouté dans TControlPosition
57	Fltfixdec	Fonctionne comme cela. Une amélioration possible : un composant dédié aux float

58	Fltpcn	Marche tout seul
59	Frcdta	Propriété frcdta sur panels. Quand on écrit le panel et que frcdta, on envoie un message windows frcdta, lorsqu'il est reçu, on affiche la fenêtre
60	Getretain	Appel à sdhUnlock
61	Help	Affichage de l'aide
62	Hlpara	Affichage de l'aide en fonction du curseur
63	Hlpbdy	Code examiné, semble ok, mais non testé à l'exécution.
64	Hlpclr	L'affichage de l'aide est dconditionné dans le programme
65	Hlpcmdkey	Fin de read sur l'écran principal ajouté. A faire : tenir compte des indicateurs sur la touche de fonction, et ne pas mettre les données dans le buffer input.
66	Hlpdoc	<b>A faire</b>
67	Hlpexcl	Pris en compte
68	Hlpfull	L'aide est toujours affichée dans la même fentre, sans importance
69	Hlpid	Voir <b>Erreur ! Source du renvoi introuvable.</b>
70	Hlppnlgrp	Affichage de l'aide
71	Hlprcd	Affichage de l'aide Reste à faire : affichage de l'aide dans un autre fichier.
72	Hlprtn	Aide non affichée (ou selon indicateur)
73	Hlpschidx	<b>Non géré</b> <b>Pas d'api pour lire les search index</b>
74	Hlpseq	Rollup_OnExecute affiche le bon format d'aide.
75	Hlptitle	Modification du titre de la fenêtre.
76	Home	Appel à sdhHome
77	Html	Pas besoin
78	Indara	Le code généré est différent. Les indicateurs sont passés au programme d'orgine via un tableau et non plus via les buffers d'entrée/sortie
79	Indtxt	Utilisé dans le designer pour aider le développeur
80	Invite	MAXDEV non compatible avec RPGOA -> invite non compatible
81	Inzinp	Save area is modified, mais pas d'exemple trouvé ou ça a des conséquences
82	Inzrcd	
83	Keep	Les formats affichés sont enregistrés dans la variable keepPanels du service programme sdsrvpgm
84	Lock	Appel à sdhUnlock..
85	Loginp	Appel à qmhsndpm pour ajouter les valeurs des zones en input dans le log
86	Logout	Appel à qmhsndpm
87	Lower	Propriété charcase
88	Mapval	Valeurs modifiées
89	Mdtoff	Mise à zéro de la propriété tag des composants ...
90	Mltchcld	Génération de plusieurs combobox
91	Mnubar	Génération d'un mainmenu
92	Mnubarchc	Génération d'un menuitem
93	Mnubardsp	Affectation d'un mainmenu
94	Mnubarsep	Voir <b>Erreur ! Source du renvoi introuvable.</b>
95	Mnubarsw	La touche de fonction associée n'est pas activée.

		Le focus n'est pas déplacé car les composant TToolButton ne peuvent pas prendre le focus. A faire : remplacer par
96	Mnucnl	La fenêtre pullDown est fermée (sdClosePullDown(...*off)
97	Moubtn	Génération de code sur evt onMouseDown ou onMouseUp
98	Msgalarm	Appel à sdBeep. Options pris en compte. Alarm pris en compte (un seul beep)
99	Msgcon	Caption modifié
100	Msgid	Modifié à l'exécution. A revoir sdRtvMsg renvoie &1 pour un caption..
101	Msgloc	Pas besoin car les messages sont affichés dans une fenêtre à part
102	Noccsid	<b>Non géré</b>
103	Openprt	Fichier prt non fermé
104	Overlay	Propriété overlay sur panel, modifié à l'exécution
105	Ovratr	Attributs des champs modifiés
106	Ovrda	Champs mis à jour selon putovr, ovrda et premier affichage
107	Pagedown /pageup	Idem rollup
108	PageUp	Idem PageDown
109	Passrcd	
110	Print	Ecran envoyé dans le spool ou indicateur de reponse mis à *on
111	Protect	Appel à sdhProtect
112	Pshbtnchc	Ajout d'un bouton
113	Pshbtnfld	Ajouts de boutons
114	Pulldown	Affiché sur click de menuitem
115	Putovr	Modification des champs (contenu et attributs) si putovr est actif (et ovrda ou ovratr)
116	Putretain	Champs en sorties : valeur envoyée uniquement si pas putretain
117	Range	Contrôlé à la validation
118	Ref	Information retrouvée, mais pas utilisée, les informations sont copiées dans l'écran.  Voir <b>Erreur ! Source du renvoi introuvable.</b>
119	Reffld	Information retrouvée, mais pas utilisée, les informations sont copiées dans l'écran.  Voir <b>Erreur ! Source du renvoi introuvable.</b>
120	Retkey/retCmdKey	Propriété retkey et retCmdKey sur panels
121	Retlcksts	Appel à sdhUnlock...
122	Rmvwdw	Dans Silverdev, les fenêtres sont systématiquement effacées.
123	Rollup	Evènement Rollup_OnExecute
124	RollDown	Evènement RollDown_OnExecute
125	Rtncsrloc	Appel à sdGetCtrl pour connaître le champ sélectionné.
126	Rtndta	Un mécanisme similaire est mis en place
127	Setof	Rien à faire, le système se démmerde tout seul Voir <b>Erreur ! Source du renvoi introuvable.f</b> Si indara : ajout de indicators(xx) = *off
128	Setoff	Idem setof
129	Sfl	Ajout d'un composant de type CSFL
130	Sflhcctl	ok
131	Sfldlt	

132	Sflcsrprg	Modification de la propriété tabulationType de la colonne
133	Sflcsrrrn	Renvoie le numéro de ligne du sous fichier ou 0 si pas de focus
134	Sflctl	Pris en compte pour la génération
135	Sflclr	Appel de sdClear sur le sfl
136	Sfldrop	Les colonnes sont toutes affichées sur la même ligne. Les colonnes qui ne sont pas sur la première ligne sont cachées / affichées sur la touche de fonction. Les colonnes sont d'abord cachées
137	Sfldsp	sdhWrite généré aussi pour le format sfl
138	Sfldspctl	Propriété DisplayFields modifiée
139	Sflend	Un label ou une image sont ajoutées. L'ascenseur vertical est mis quel que soit la valeur de *srcbar
140	sflenter	La dernière ligne est sélectionnée
141	Sflfold	Les colonnes sont toutes affichées sur la même ligne. Les colonnes qui ne sont pas sur la première ligne sont cachées / affichées sur la touche de fonction. Les colonnes sont d'abord affichées
142	Sflinz	Des lignes vides sont ajoutées
143	Sflin	Le sous fichier est affiché en lignes. Le nombre de colonnes est identique à la version 5250
144	Sflmltchc	Ajout d'un sous fichier avec deux colonnes. Première colonne de type csBoolean
145	Sflmode	Renvoie toujours 0
146	Sflmsg	Message affiché
147	SflMsgld	Idem SflMsg
148	Sflmsgkey	Ajout d'un sous fichier avec lecture dans programme message q
149	Sffmsgrcd	Ajout d'un sous fichier avec lecture dans programme message q
150	Sflnxtchg	Pas d'appel à sdReinit dans fonction update si sflNxtChg
151	Sflpag	Pas d'influence sur le programme ou l'écran
152	Sflpgmq	Ajout d'un sous fichier avec lecture dans programme message q Reste à voir sflPgmq de taille 276
153	Sflrcdnbr	Ajout de sdGotoCell
154	Sflrna	sdReinit appelé si sflrna et sflinz
155	Sflrolval	Propriété pageHeight
156	Sflrttsel	Propriété SFLRTNSEL ajoutée
157	Sflscroll	Appel à sdGetInt sur topRow
158	Sflsiz	Pas d'influence sur le programme ou l'écran
159	Sflsngchc	Ajout d'un sous fichier avec deux colonnes, première colonne de type csRadio
160	Sln	Propriété sln sur panels, controles déplacés
161	Sngchcflld	Création d'un composant CRadioGroup
162	Sysname	Appel à sdGetSysName
163	Text	Pas d'incidence
164	time	%char(%Time)
165	Timfmt	Pris en compte pour la déclaration des buffers. (obligatoire)
166	Timsep	Pris en compte pour la déclaration des buffers. (obligatoire)
167	Unlock	Appel à sdhUnlock
168	User	Appel à sdGetCurUser
169	Usrdfn	<b>Non traité automatiquement, l'utilisateur peut modifier le handler et</b>

		<b>créer les champs comme en silverdev classique</b>
170	Ussrdspmgmt	La fenêtre est rendue transparente en modifiant ses propriété transparentcolor et transparentcolorvalue
171	Usrrstdsp	
172	Valnum	Si valnum, minvalue est à 0
173	Values	Composant CComboBox créé
174	Vldcmdkey	Indicateur mis à *on dans actions
175	Wdwborder	Rien à faire
176	Wdwtitle	Ajout du titre dans la fenêtre
177	Window	Ajout d'une seconde fenêtre en modal.
178	Wrdwrap	Pas d'influence car champs sur une seule ligne