

Silverdev conversion

I.	Introduction.....	3
A.	Technology	3
B.	Release	3
C.	Objects.....	3
D.	Terminology.....	3
II.	Conversion.....	4
A.	Architecture.....	4
B.	Example	5
C.	Properties window	12
III.	Call the converted program.....	14
A.	CL program	14
B.	Icon	15
IV.	Compatibility with classic Silverdev	16
A.	Calls	16
B.	modal forms	19
C.	Call of a classic silverdev program from a handler type program.....	20
D.	starting program.....	20
V.	Improvements during conversion	20
A.	Click on closing cross.	20
B.	Fields.....	20
C.	Deactivation of function keys or fields.....	24
VI.	Improvements post conversion.....	24
A.	Introduction.....	24
B.	Modification in the handler.....	24
C.	Modification in the copied program	40
D.	Modification in the handler and the copied program.....	40
VII.	Regeneration	44
A.	Introduction.....	44
B.	Save properties.....	45
C.	Reloading properties	45
D.	Regenerate menu	46
E.	Save manual modifications	46
F.	Removing components.....	48
G.	Protection	48

VIII.	Set default values	49
A.	OnClose function key.....	51
B.	Conversions	52
C.	SFL.....	52
IX.	Constraints	52
X.	RPG 3 programs.....	52
XI.	Replacement solutions.....	53
A.	Calling a system program	53
B.	Command line	53
C.	Dsply	53
D.	Deleting	53
A.	RECVF.....	53
XII.	Reference fields.....	54
XIII.	Licences	54
XIV.	Several screens in a program	55
XV.	Habitual mistakes	55
XVI.	Full example	56
A.	Présentation	56
B.	Context creation	56
C.	Handlers generation	57
D.	Copy origin programs	58
E.	Starting converted program	58
F.	Modifications in generated programs.....	63
XVII.	Multiple generations	65
XVIII.	KeyWords	66

I. Introduction

A. Technology

Silverdev conversion tool uses RPGOA technology.
Thus, there are some constraints. See chapter on constraints.

B. Release

Since Silverdev conversion tool is based on RPGOA technology, minimum operating system is V6R1.
Conversion tool is included in Silverdev from release V3R8 of Silverdev.

C. Objects

The origin application contains a rpg program(*pgm) and a 5250 screen (*FILE/DSPF)
To convert a program, you need RPG sources of the program.
For the screen, DSPF object is enough.

Silverdev tool designer allows to generate a handler for the screen.
A handler is a RPG program that will be called each time the main program uses the screen.
A Silverdev screen is also generated. This Silverdev screen is used by the handler.

Warning : For those used to classic silverdev development, the generated screen by silverdev is different from those you know. It includes windows and records.

The origin program will be copied, and the declaration of the screen will be modified in this copy.
The instruction handler('HANDLER') will be added in the screen declaration.
Copied program will be compiled.

Summary:

5250 version	Silverdev version (converted)
Origin program	Copied program
5250 screen	5250 screen Handler program Silverdev screen handler

D. Terminology

In this document, following terms will be used :

Origin program	Source and object of the 5250 rpg program that is converted.
----------------	--

Copied program	Source and object of the copied (and modified) rpg program.
Handler program	Source and object of the generated program used as a handler.
Silverdev screen handler	Source and object of the generated screen. This screen is used collectively with the handler program.
5250 screen	Origin screen. This screen must exist and in the library list when executing the converted program.

II. Conversion

A. Architecture

For a program named pgm1 in the library lib1, several architectures are possible, we will chose this one :

Sources of origin program will be copied in a library called lib2 and compiled in lib2.
Compiled program will not be renamed. It's important because it can be called by another converted program.

Handler program and its source are generated in lib2.
Handler program can have the name of the screen if this one is different from the origin program.
When 5250 screen has same name as origin program, handler program must have another name.
Silverdev screen will be generated in lib2. (object of type *USRSPC, source will be in ifs)

At the end, we will have

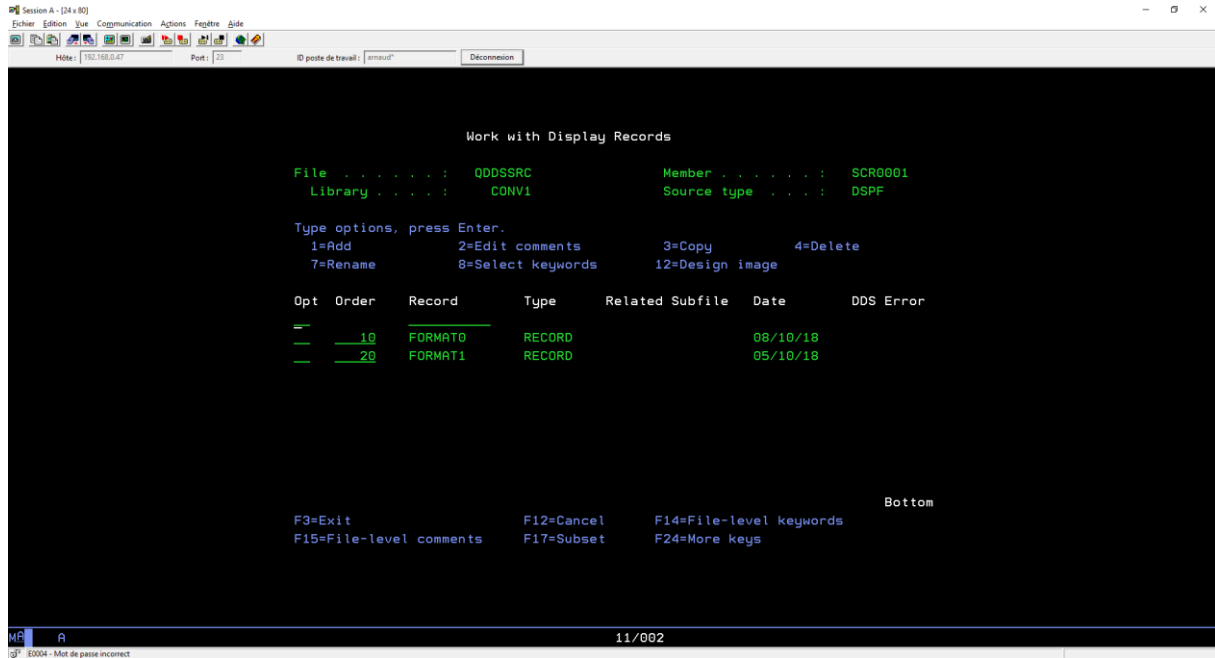
Srclib1	Lib1	Srclib2	Lib2	ifs
Pgm1 sources	Pgm1 (*PGM)	Source of pgm1 (copied)	Pgm1(*PGM)	Silverdev screen source handler1
	Screen1 (*FILE/DSPF)	Source of handler1	HANDLER1(*PGM)	
			HANDLER1(*USRSPC)	

Note 1 : It is possible to keep only one source, by putting pre compilation instructions around screen declaration.

Note 2 : In examples of this document, sources and objects will be in the same library. SRCLIB1 and SRCLIB2 will not exist.

B. Example

In the following example, we have program conv1/pgm0001 and screen conv1/scr0001



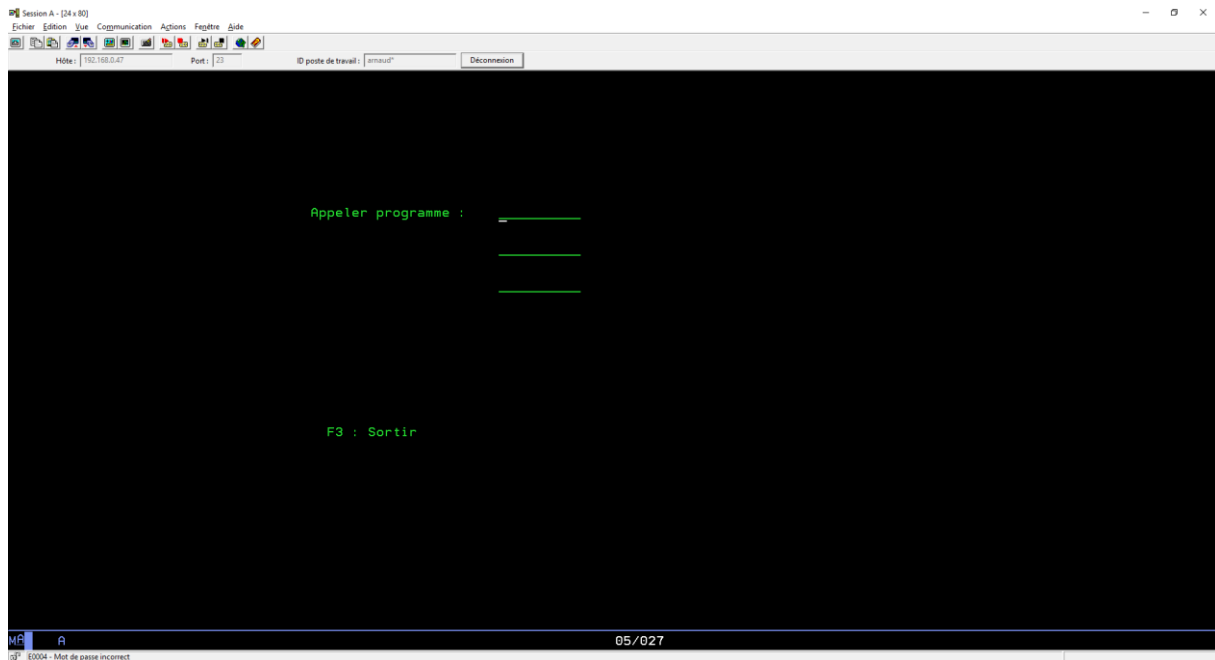
```

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

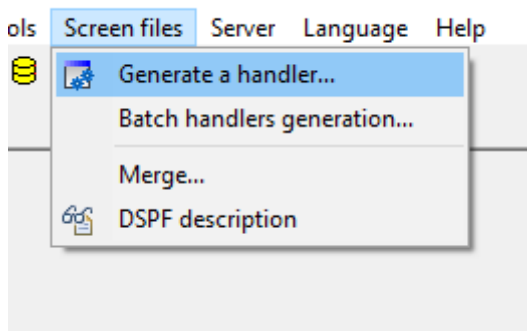
```



A Silverdev context is created in CONV2. (See Silverdev documentation to know more about contexts)

Handler Generation :

Use menu "Screen files/ Generate a handler"



Fill in the fields as indicated on the following screen capture :

Handler generation

5250 screen

Library : CONV1 ...

Name : SCR0001 ...

Silverdev program

Context : CONV2 ...

Program : HDL0001

Screen : *PGM

Type : SVDHRPG

Description : Handler for CONV1/SCR0001

OK Cancel

A window will appear, leave the pre set values, we will come back later on this window.

Click on validate

CONV1/SCR0001

Silverdev form

Width :800

Height :600

Margin :20

FORMAT0 *DS3

FORMAT1 *DS3

Record type

RECORD

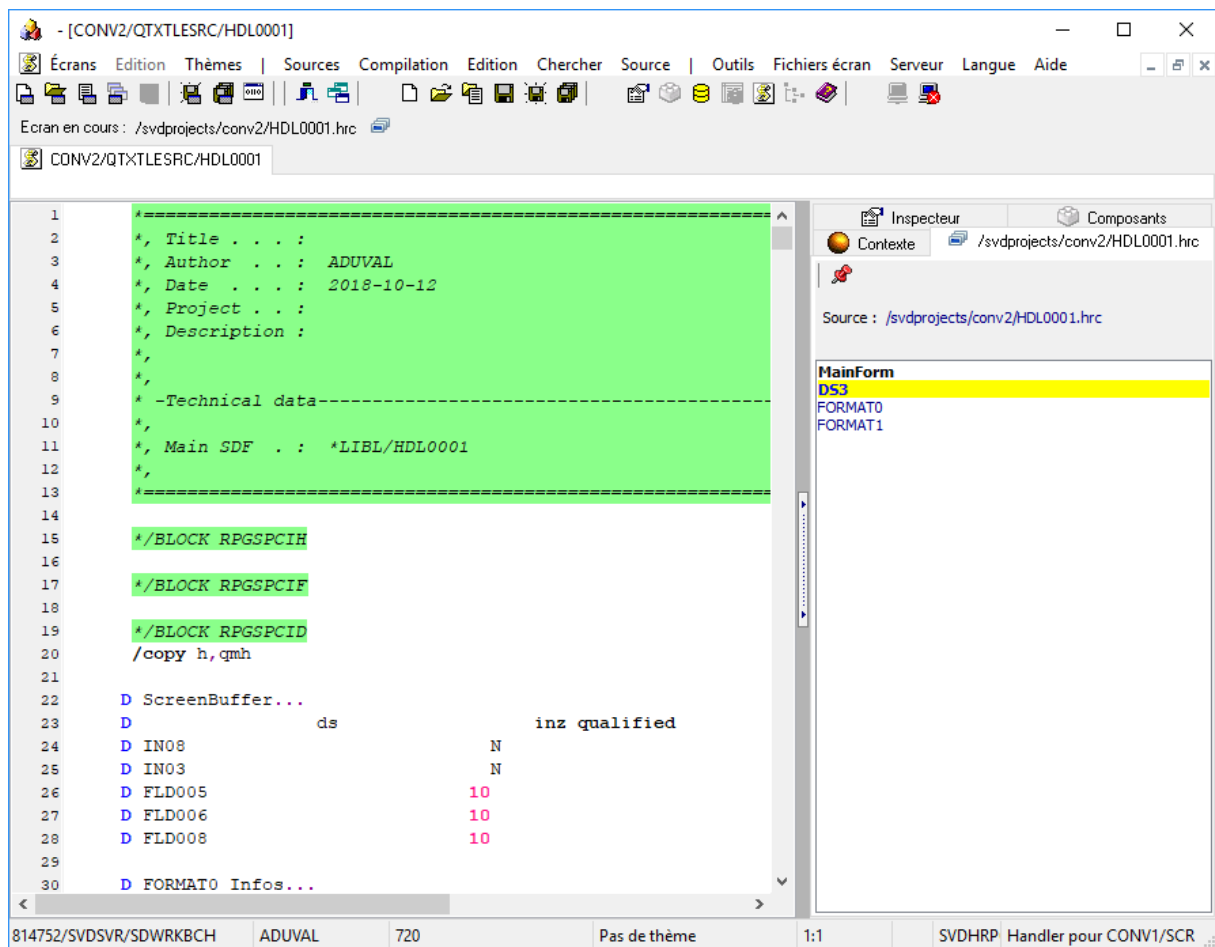
>> Function keys

>> Fields

Validate

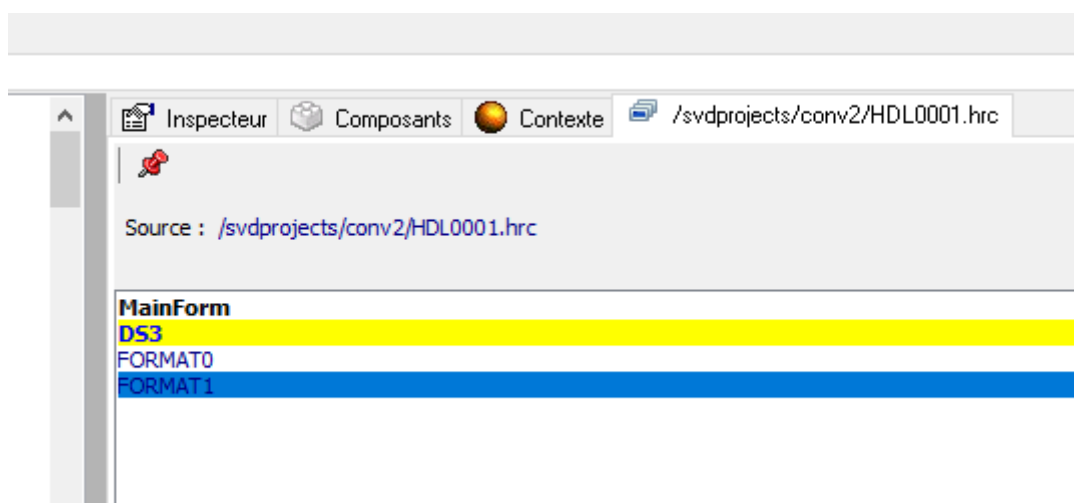
Cancel

Generated rpg source and generated screen source are displayed :

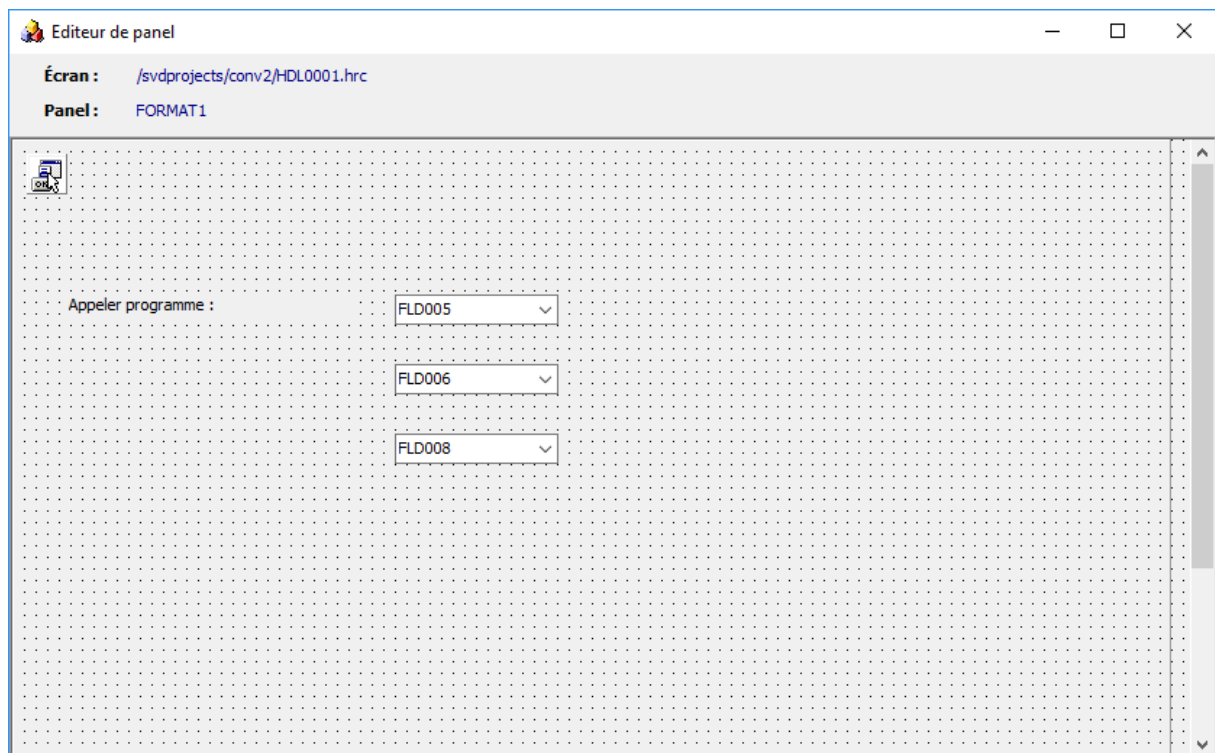


Handler program is generated, next steps are not mandatory.

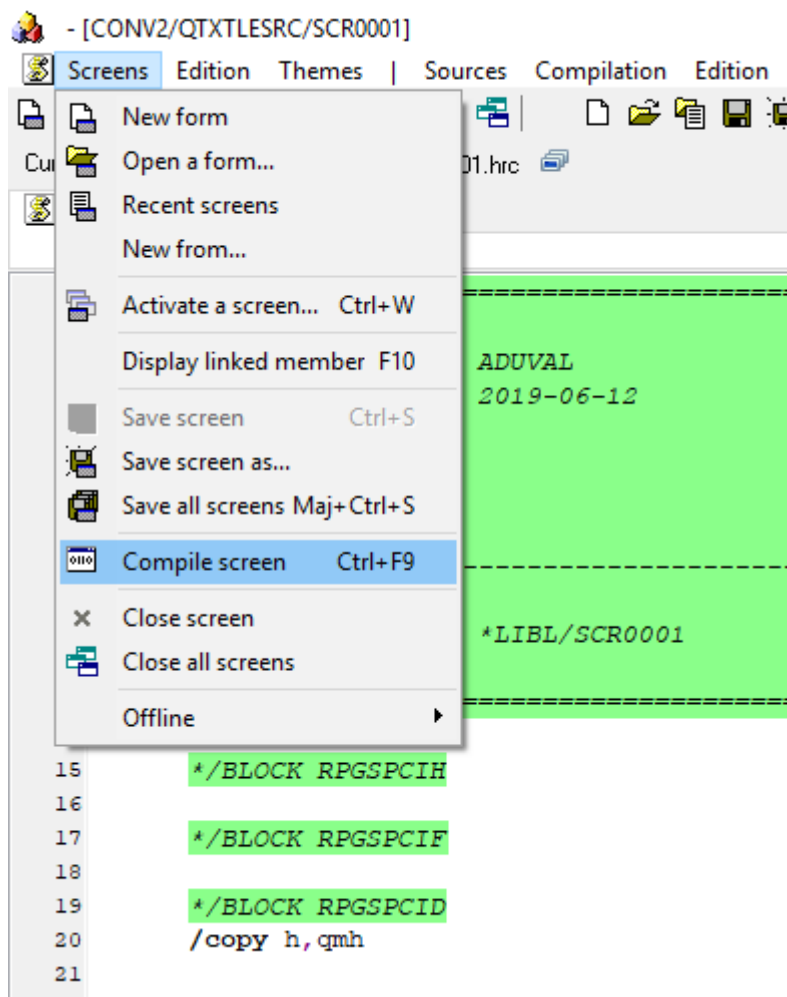
Figure 1 To change a record on silverdev screen, double click on its name.



Some modifications can be done, See chapter on improvements of generated programs.



If you modify the screen, you have save and compile it.



C. Properties window

During conversion, a properties window is displayed :


CONV1/AAA

Silverdev form

Width : 800

Height : 600

Margin : 20



FORMAT0 *DS3

Record type RECORD

>> Function keys

>> Fields

Validate

Cancel

The first part allows to determine height and width of the window that will be used to display data on screen.

Components will be place proportionnaly on the screen.

Example :

When a field is on line 2 and column 10, for a display of 24X80, the component will be in position :

left : $10 + 2/24 * 600$

top : $10 + 10/80 * 800$

13

CONV1/SCR0001

Silverdev form

Width :

Height :

Margin :

FORMAT0 *DS3 FORMAT1 *DS3

Record type RECORD

Function keys

OnClose, function key CACF03 03

Key function	Sortable
CACF03	<input checked="" type="checkbox"/>
CACF08	<input type="checkbox"/>
ENTER	<input type="checkbox"/>

>> Fields

Validate Cancel

For each record , there is a tabsheet.

You can in the tabSheet, indicate which function keys are activated when user click on closing cross. F3 and F12 function keys are pre checked. Unchecked them if theses function keys are used for something else.

We will come back on the properties window.

III. Call the converted program

A. CL program

We start by creating a cl program that will set the correct library list :

```

pgm
ADDLIB LIB(CONV2) POSITION(*AFTER QTEMP)
MONMSG MSGID(CPF0000)

ADDLIB LIB(CONV1) POSITION(*AFTER CONV2)
MONMSG MSGID(CPF0000)

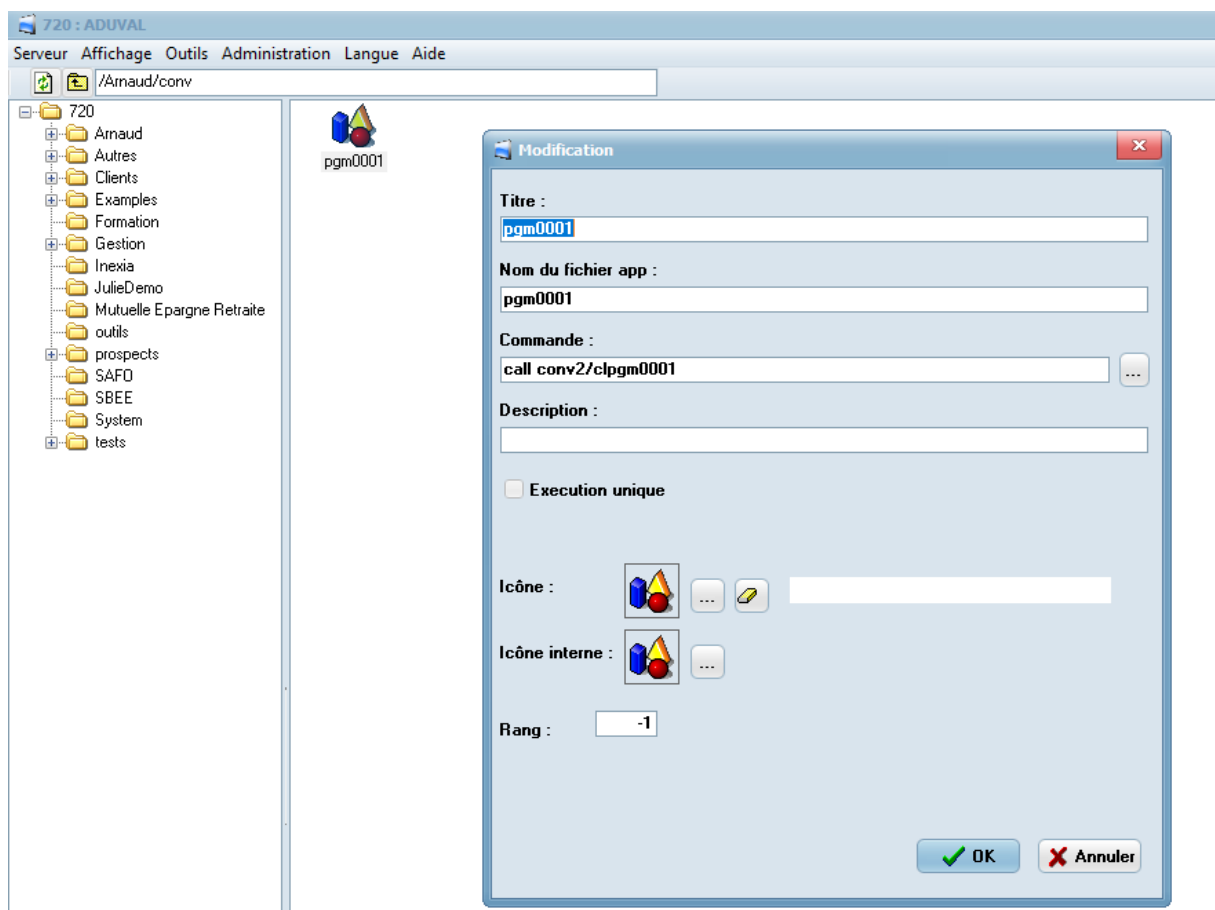
CALL PGM(*LIBL/PGM0001)
endpgm

```

The library of the 5250 screen is in library list because 5250 screen must be in library list at runtime.
Library with created objects is put in first position.
The converted program is called.

B. Icon

A new icon is created in MyDesk.



Note : MyDesk is the program used to start Silverdev applications.
See general documentation to have more informations about MyDesk.

IV. Compatibility with classic Silverdev

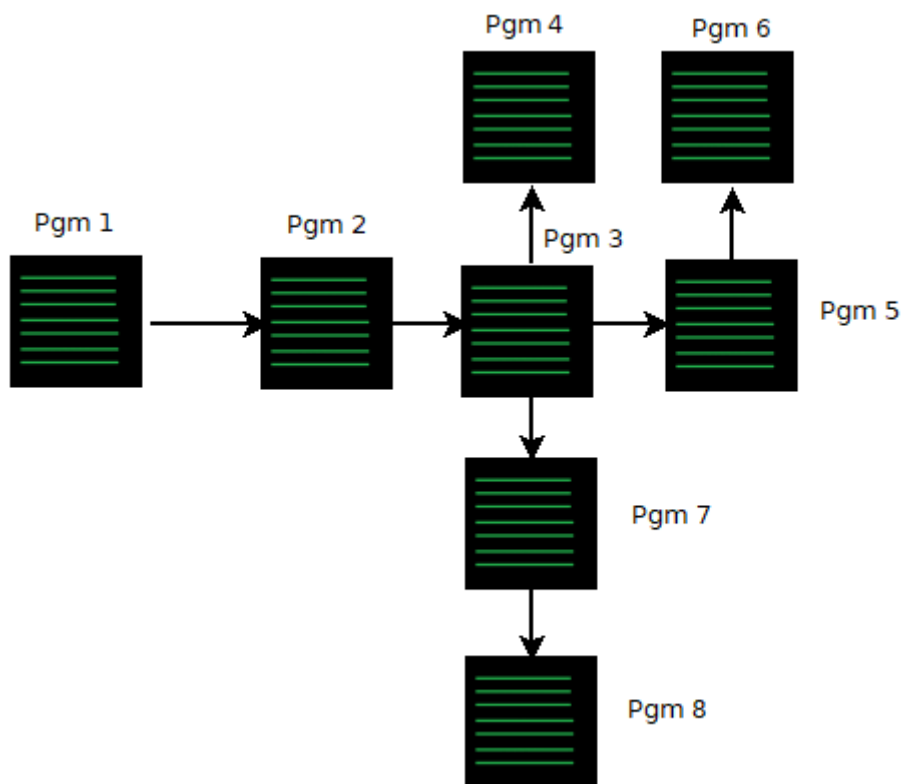
A. Calls

One of the strong points of Silverdev conversion tool is compatibility between converted programs and classic silverdev programs.

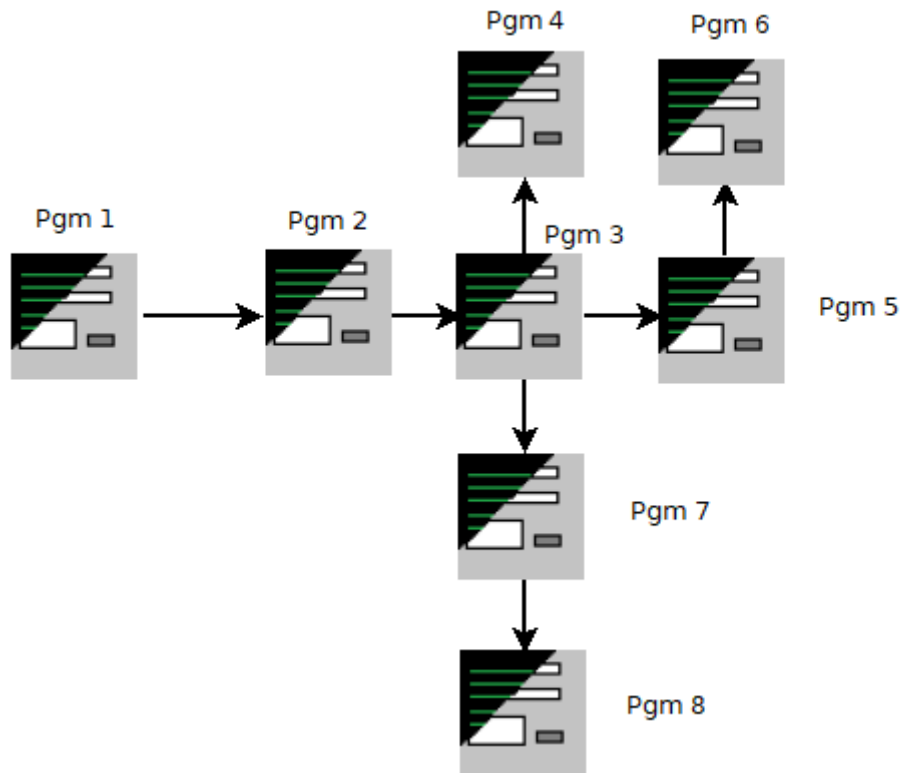
This means that a Classic Silverdev program can call a converted Silverdev program and vice versa.

You can convert a chain of programs forming an application and replace step by step converted programs by classic silverdev programs

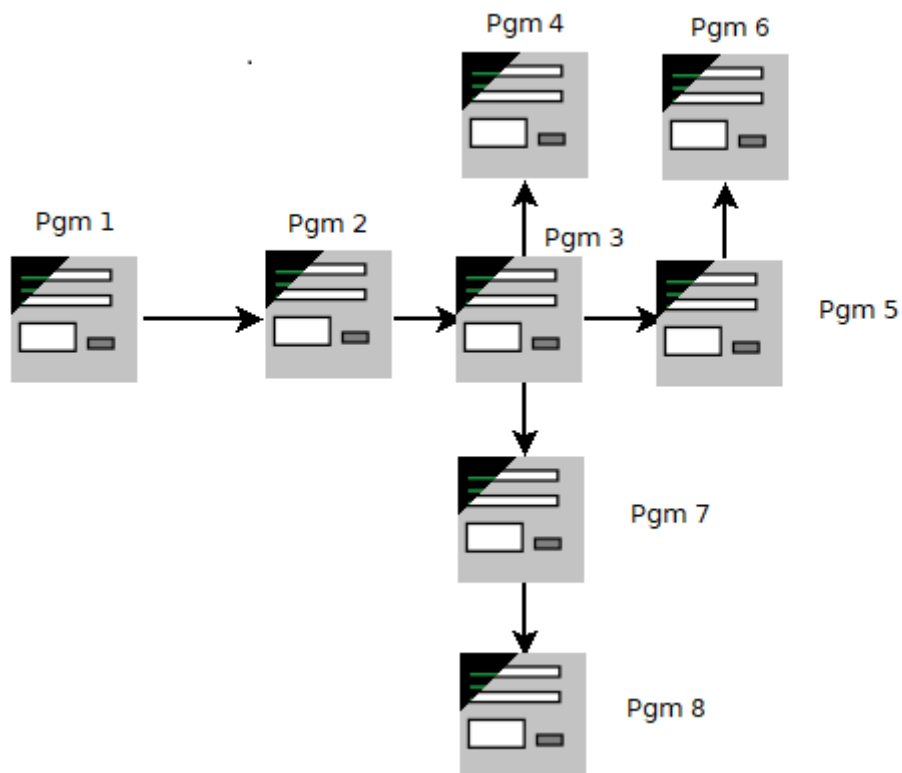
Having the followin program chain :



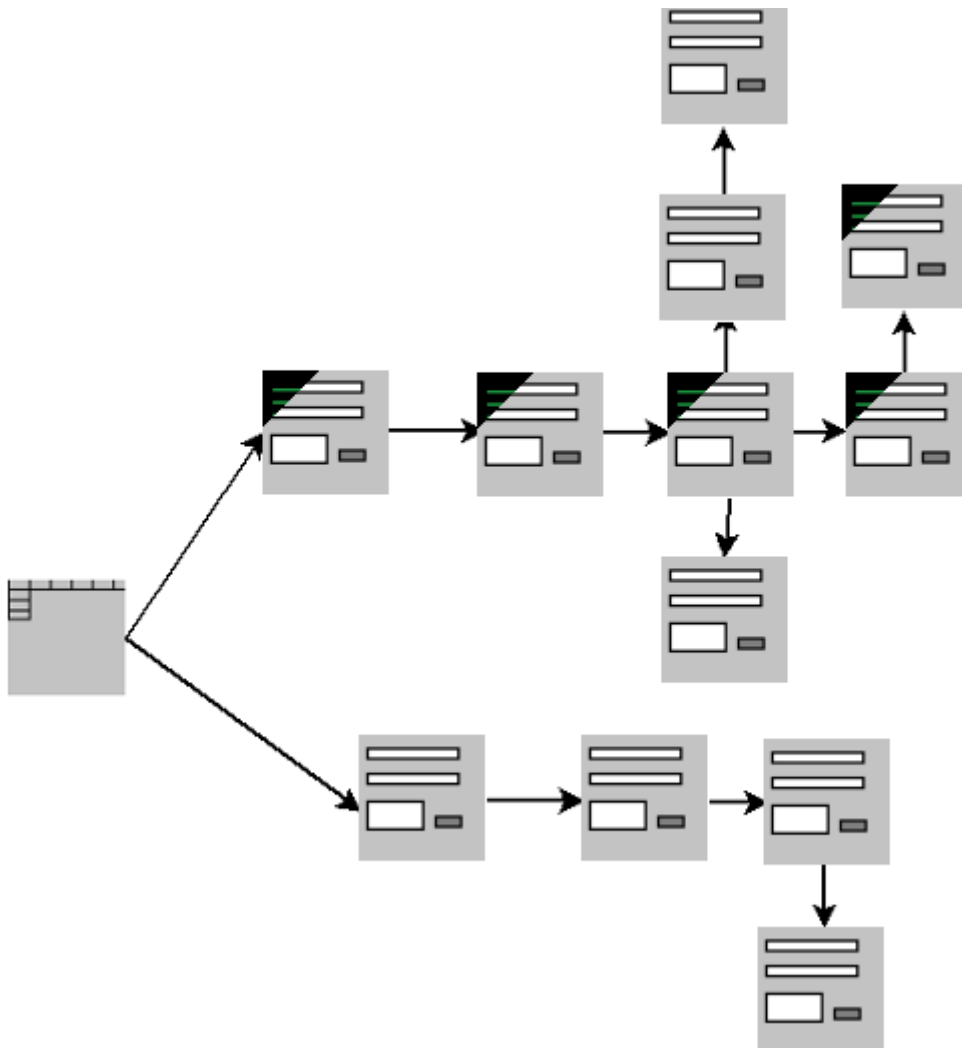
After creating handlers , we have the following chain :



You can then improve the generated programs

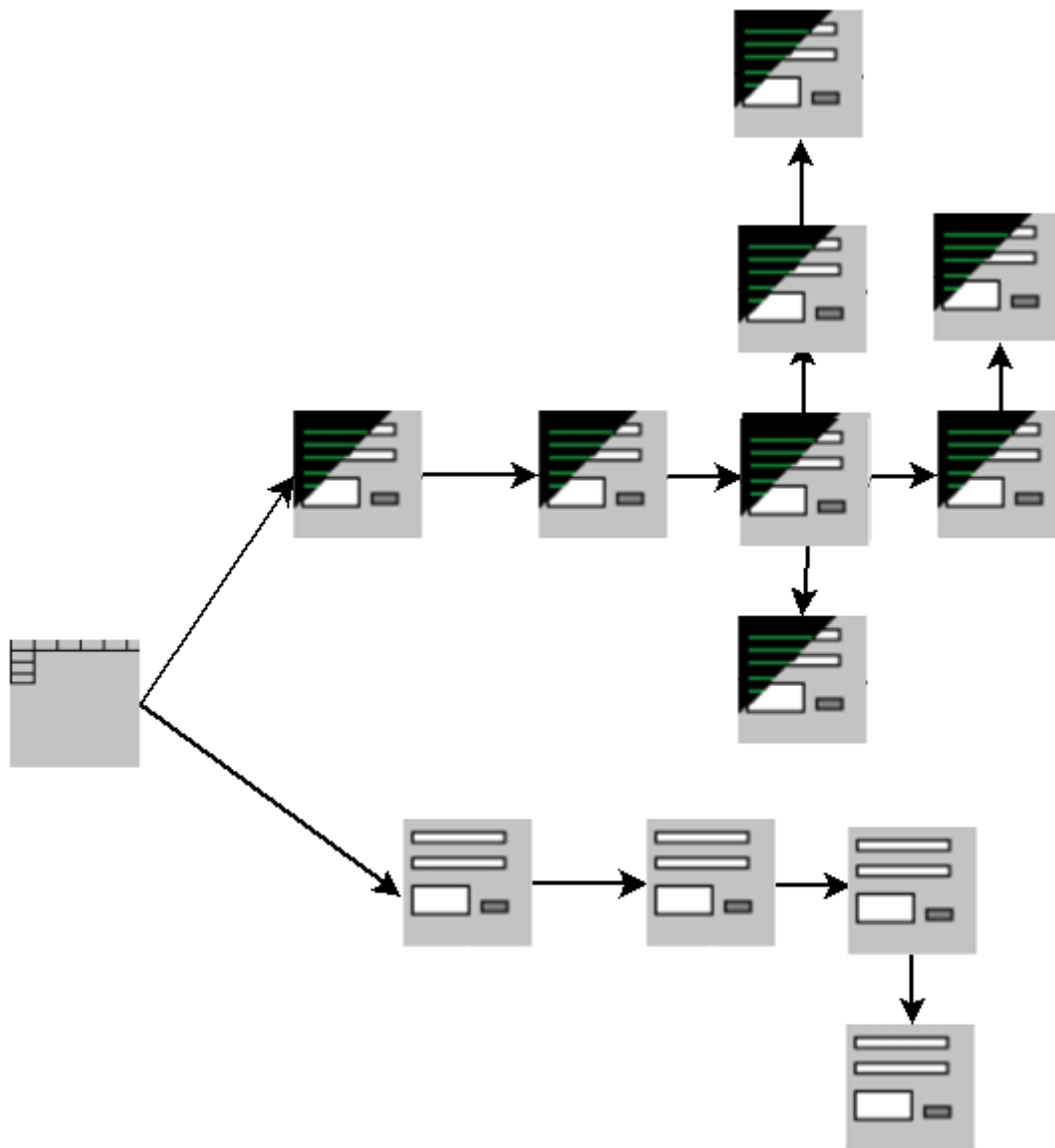


You can then replace some of the programs by classic silverdev programs :



In following example, programs 7 and 8 have been re made with classic Silverdev and have been fusionned. Program 4 has been re made with classic silverdev, and now calls a new program, program 9 , done with classic silverdev.

Then, it's interesting to create a classic silverdev program, at the root of the program chain, having a menu to call converted programs or new classic silverdev programs.



B. modal forms

To have a similar behavior to 5250 programs, but above all to respect 5250 programs ergonomics, converted silverdev programs always use modal forms.

Each 5250 screen converted has its own form.
This allows compatibility with classic silverdev programs.

If for example, a program chain is composed of pgm1, pgm2, pgm3 as follows :

Converted program	Classic silverdev program
Pgm1	
	Pgm2
Pgm3	

It is possible that pgm1 calls pgm2 and pgm2 calls pgm3.
That would not be possible if all the converted programs was displayed in a same form.

C. Call of a classic silverdev program from a handler type program.

The classic silverdev program must be modal, otherwise, form of classic program is going behind the handler program form.

D. starting program

In order to make classic silverdev programs and converted programs cohabit, a simple solution is to create a starting program with a menu, in classic silverdev, with a CMainMenu component. From this menu, classic programs and converted programs can be called.

V. Improvements during conversion

A. Click on closing cross.

When generating a handler, a window is displayed.

It allows to select a function key that is activated when clicking on the closing cross.

By default, F3 and F12 function keys are checked.

This allows to user to exit the window by pressing F3 as the program allows or to click on the closing cross of the window.

If no function key is checked, the cross will not close the window.

B. Fields

1. 5250 graphic

A graphic on the right displays approximatively the 5250 screen record to retrieve fields.

The field selected in the grid is displayed in yellow.

You can also click on the graphic to select a field in the grid.

CONV1/SCR0175

Silverdev form

Width : 800

Height : 600

Margin : 20

RECORD1 *DS3

Record type RECORD

Name	Type	Attribute	Component	Omit
NUM8	Signed numeric zoneBoth		CDateEdit	<input type="checkbox"/>
NUM6	Signed numeric zoneBoth		CDateEdit	<input checked="" type="checkbox"/>
NUM8B	Numeric only 8	Both	CDateEdit	<input type="checkbox"/>
TOTO	Alpha shift characterBoth			<input type="checkbox"/>

NUM8
NUM6
NUM8B
TOTO

Validate Cancel

When two fields overlap, (conditionned on indicators) only one of the fields is displayed. Nevertheless, the selected field in the grid is always displayed.

2. Component Type

In the “component” column, you can select the kind of component used for the field. If you leave the empty value, the designer will select a component type automatically.

3. Potential date/time fields

If you select CDateEdit component for a numeric value, a form is displayed to select date format in the field :

CONV1/SCR0175

Silverdev form

Width : 800

Height : 600

Margin : 20

RECORD1 *DS3

Record type RECORD

Name	Type	Attribute	Component	Omit
NUM8	Signed numeric zoneBoth		CDateEdit	<input type="checkbox"/>
NUM6	Signed numeric zoneBoth		CDateEdit	<input checked="" type="checkbox"/>
NUM8B	Numeric only 8	Both	CDateEdit	<input type="checkbox"/>
TOTO	Alpha shift characterBoth			<input type="checkbox"/>

Date Edit

Conversion :

YYMMDD
YYDDMM
MMYYDD
MMDDYY
DDYYMM
DDMMYY
CYMMDD

Cancel

Validate Cancel

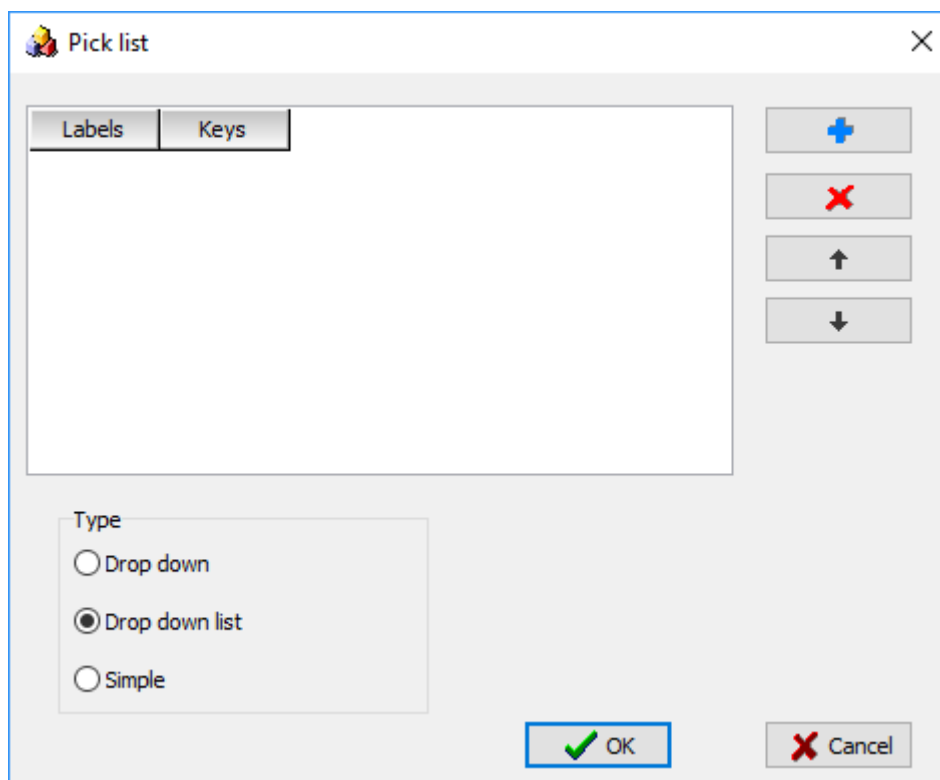
Note , the CDateEdit component type can be selected automatically by configuring the context. See VIII.B

4. Pick list

In order to generate a picklist for a field, you can select the value picklist in the improvement column.

Attribute	Improvement	Header image	
Constant	Pick list	▼	In
Constant			In
Constant			For

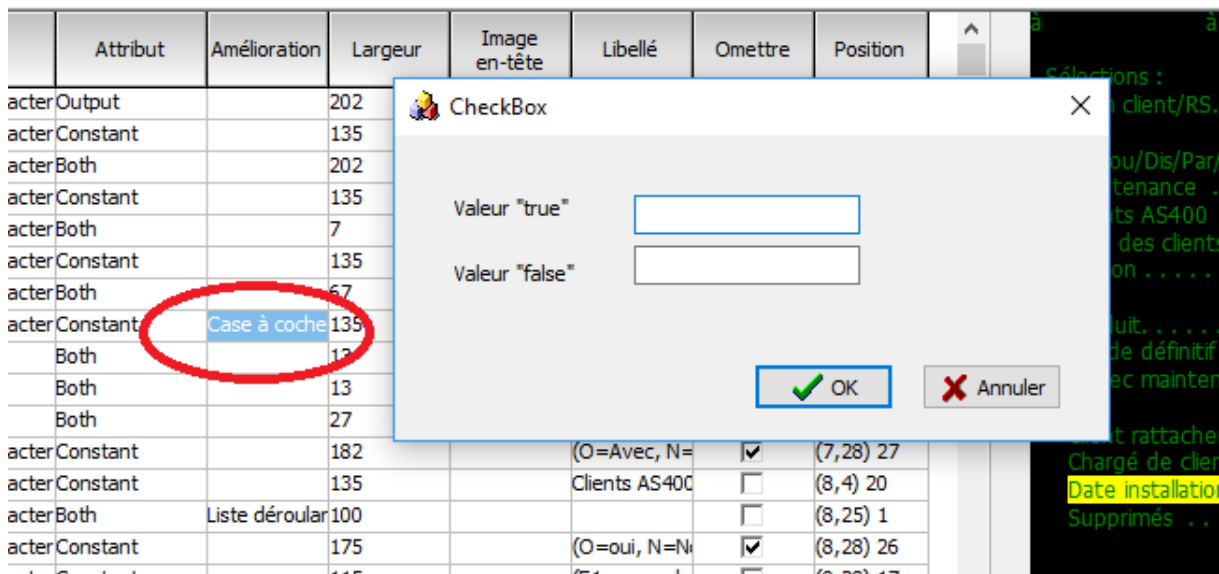
A form is displayed to build the pick list



The image shows a dialog box titled "Pick list" with a close button (X) in the top right corner. Inside the dialog, there is a large rectangular area divided into two tabs: "Labels" and "Keys". To the right of this area are four buttons: a blue plus sign (+), a red minus sign (-), an upward arrow (↑), and a downward arrow (↓). Below the main area, there is a section labeled "Type" with three radio button options: "Drop down", "Drop down list" (which is selected), and "Simple". At the bottom of the dialog are two buttons: "OK" with a green checkmark icon and "Cancel" with a red X icon.

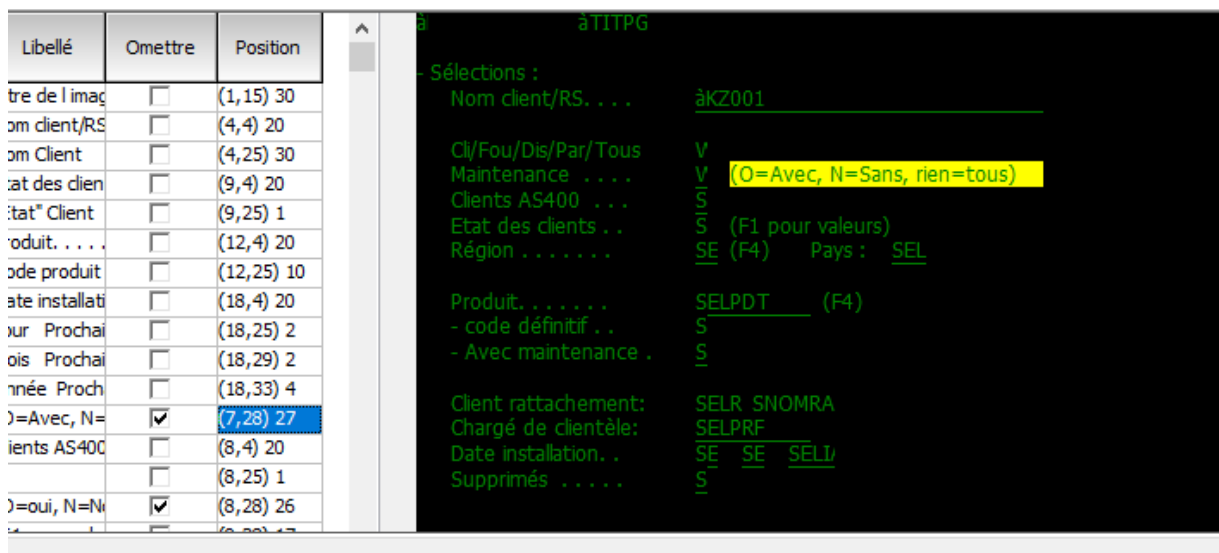
5. Check box

If you pick "Checkbox" in column "improvement", a dialog box is displayed to set the values read by the program depending on state of the checkbox.



6. Omit fields

To omit a field, check the checkbox in "omit" column.
Checked Fields in "omit" checkbox are not displayed in the graphic.



When a field is omitted, it is not displayed anymore in the 5250 graphic.
(Except if it is selected in the grid)

C. Deactivation of function keys or fields

Sometimes, you will prefer to deactivate a function key or not display a field.

For example, if the function key F9 displays a command line, you can deselect this function key and remove the text displaying "F9=command line"

The image shows two configuration windows. The top window, titled 'Fields', contains a table with the following data:

Field	Type	Numeric conversion	image en tête	Label	Omit
FLD001	Alpha shift character				<input type="checkbox"/>
Valeur saisie :	Alpha shift character			Valeur saisie :	<input type="checkbox"/>
FLD002	Alpha shift character				<input type="checkbox"/>
F9 = ligne de commande	Alpha shift character			F9 = ligne de	<input checked="" type="checkbox"/>

The bottom window, titled 'Function keys', has a dropdown menu 'OnClose, function key' set to 'CACF03 03'. Below it is a table with the following data:

Key function	Omit
CACF03	<input type="checkbox"/>
CACF09	<input checked="" type="checkbox"/>
ENTER	<input type="checkbox"/>

VI. Improvements post conversion

A. Introduction

copied program and handler program follow the original algorithm.

So ergonomics cannot be modified entirely.

If you want to re-think entirely program ergonomics, it's better to re-write the program with classic Silverdev

You can change the generated program (copied program + handler program) to improve the result

B. Modification in the handler

1. Introduction

Handler can be modified with classic Silverdev tools.

Warning, window handle is stateInfo.F1 and components name are qualified.

Don't write any code between tags SvdGenerated++ and SvdGenerated--, or your code would be lost.

Code between these two tags is greyed so you are not tempted to change it :

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

D setInIndicators...
D                                pr
D aFormat                      10      value

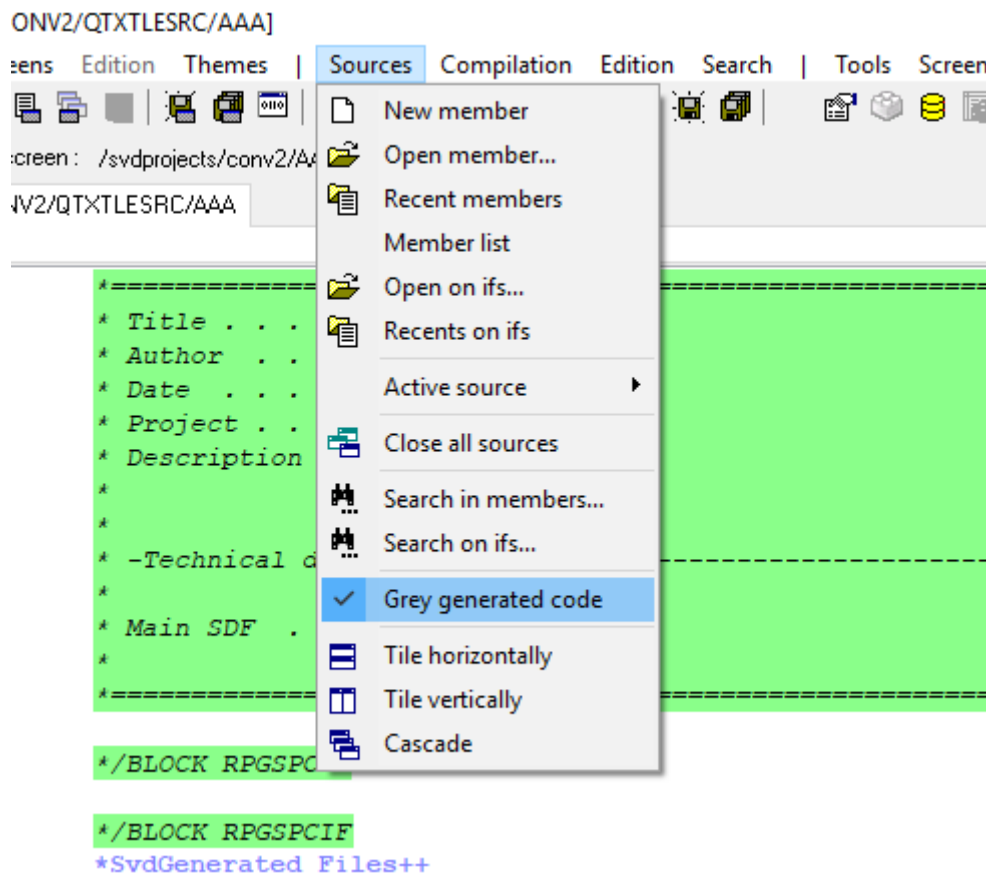
D changeInIndicators...
D                                pr
D aInd                          2  0  value
D aVldCmdKey                     N      value

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
```

You can decide not to grey this code with the following menu item :



2. Properties modifications

Components properties can be modified.

For example, size and position can be modified.

These modifications can be saved in case of re generation. (See Re generation chapter)

3. Replacing components

A generated component can be replaced by another . In this case, you have to fill the blocks CUSTOMREAD and CUSTOMWRITE

a) Example 1 :

Three numeric fields are replaced by a DateEdit component :

The following code allows to pass value of dateEdit to screen buffer :

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);
```

CustomTest block allow to add a test on the read value :

*/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) Example 2

5250 screen is like that :

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

Your choice : —

F3=Fin

Select 'CRadioButton' as component, set the key property to 1,2,3... and remove the field « Your choice »

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

In CustomRead block write the following code :

```

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

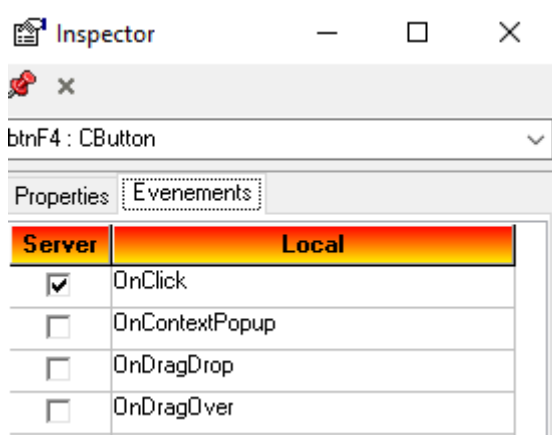
```

4. Adding a button

In the program, the F4 key allows to display a selection list.
 We are going to add a button that will allow to display this list.



For the button, check the OnClick event :



,then double click on the OnClick event line.

Write the code that will run when the user clicks on the button.

We call the CACF04_ONEXECUTE function, but before, we set the focus, in the field because the CACF04_ONEXECUTE function tests wich field is focused.

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

5. Adding an image

Adding an image in the screen is possible, just add a component CImage in the panel corresponding to the record. The image will be added to the screen when the panel is written.

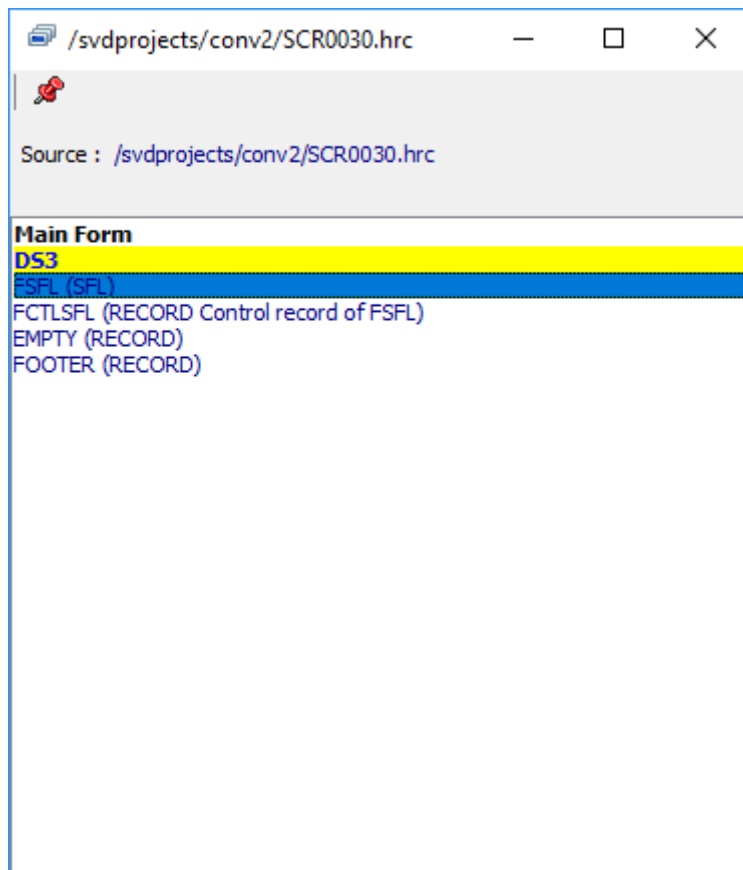
Components can be added directly on the main window or on the records.

6. Add a double click on sfl

If in the 5250 version , it is possible to select a line by entering an option, you can add the possibility to double click to select quicker.

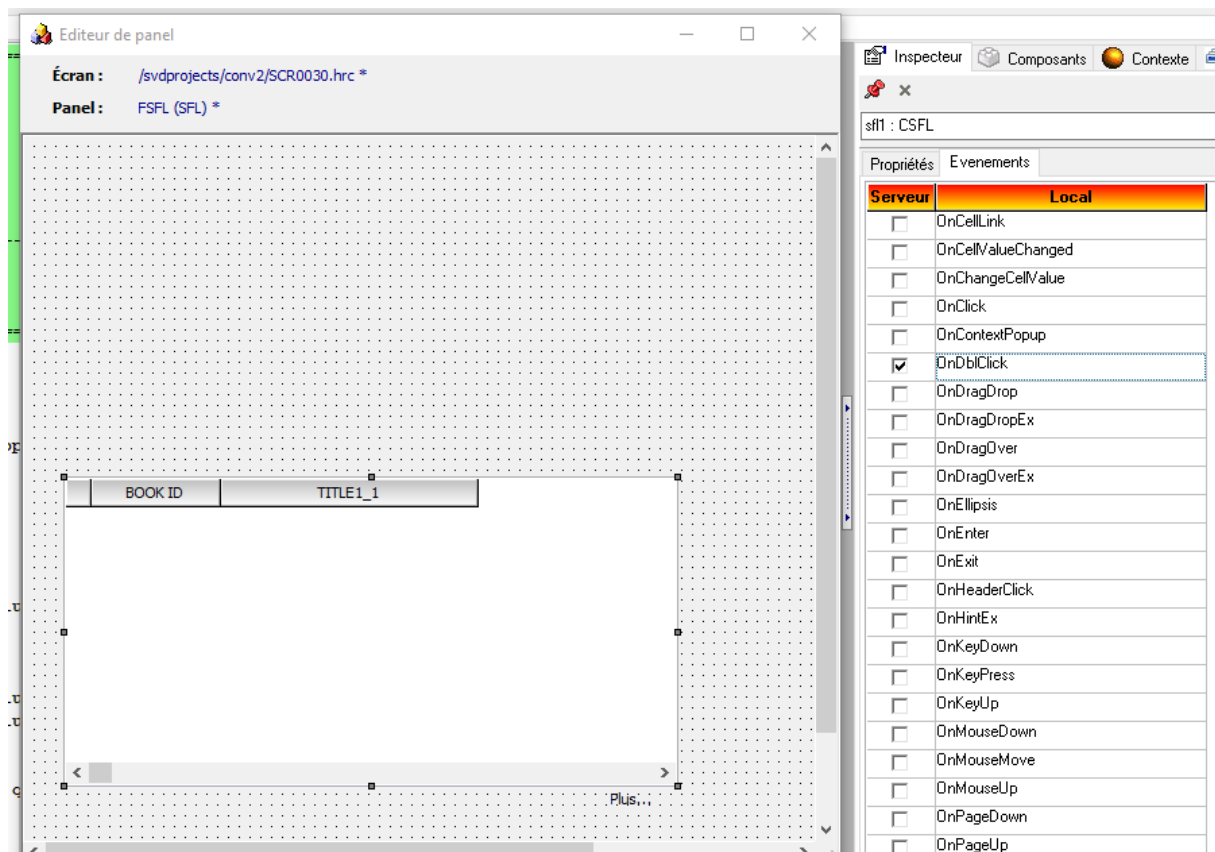
a) Open panel

In generated screen, double click on the sfl format :



b) Double click

Select the CSFL component, and in the inspector, check OnDbClick event :



c) Rpg code

Double click on “OnDbClick” row

Cursor is inserted in rpg code :

```

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

```

Enter the following code :

```

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

```

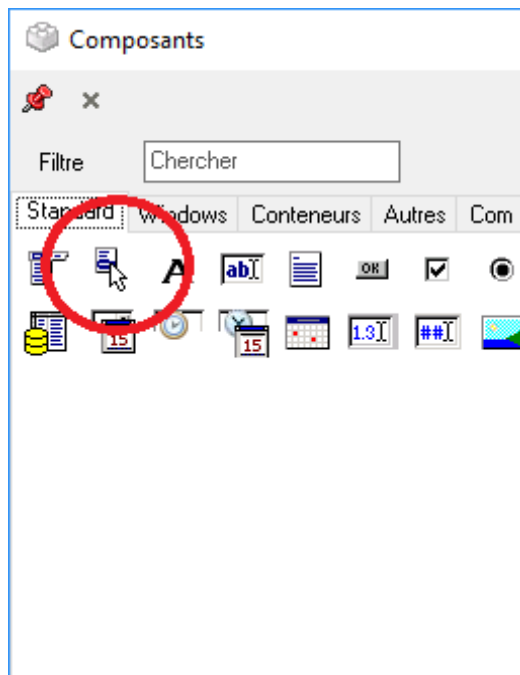
This will insert the value “1” in the option column and simulate the enter key.

7. Add a popup menu to sfl

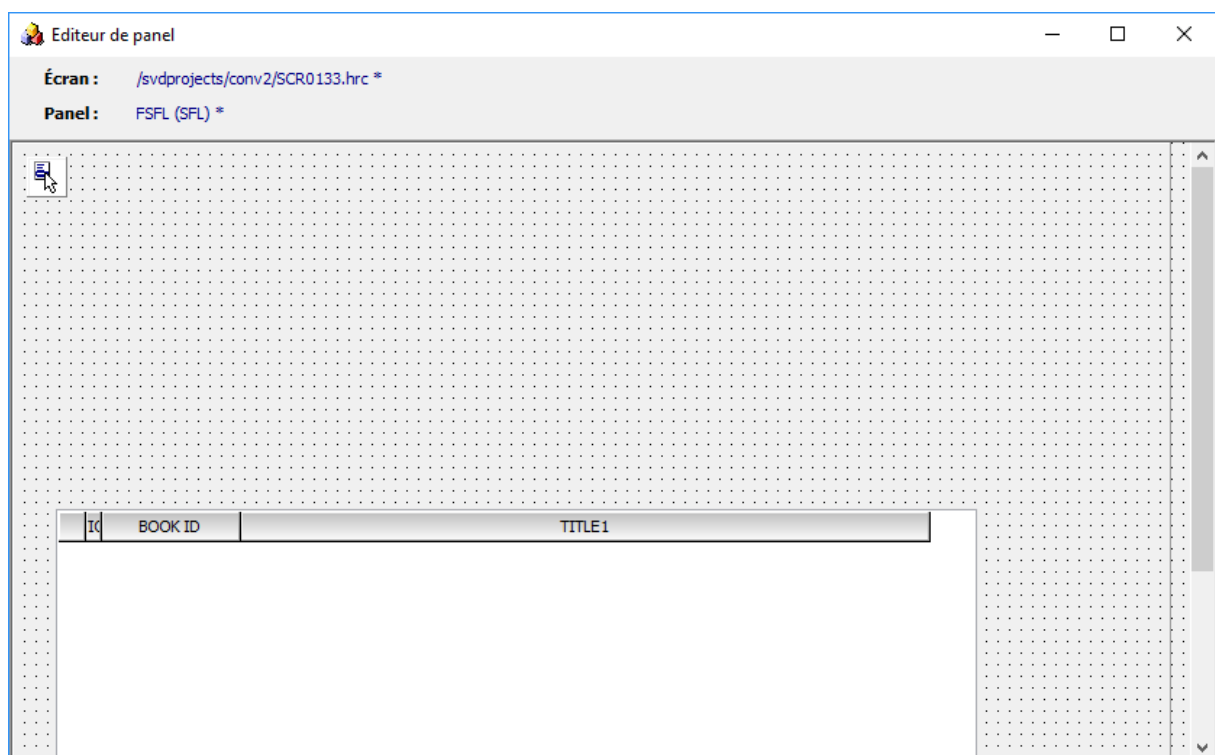
When several options are authorized on a sfl, it can be interesting to add a popup menu that will simplify these options.

a) CPopupMenu component

Select the CPopupMenu component in the component palette.



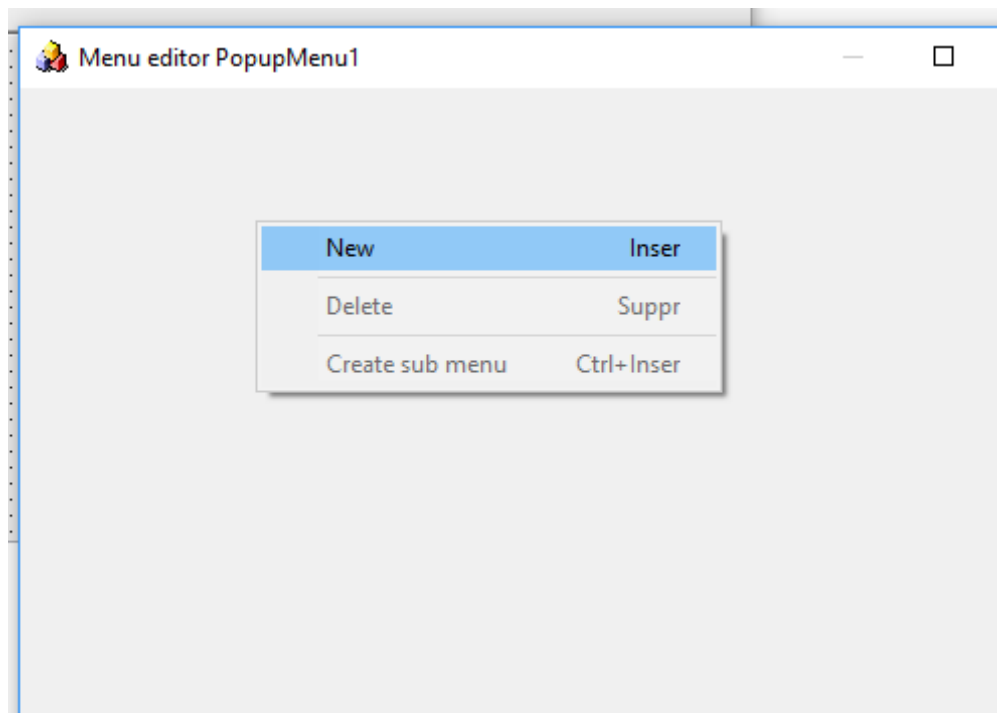
Drop the component on the panel containing the CSFL component



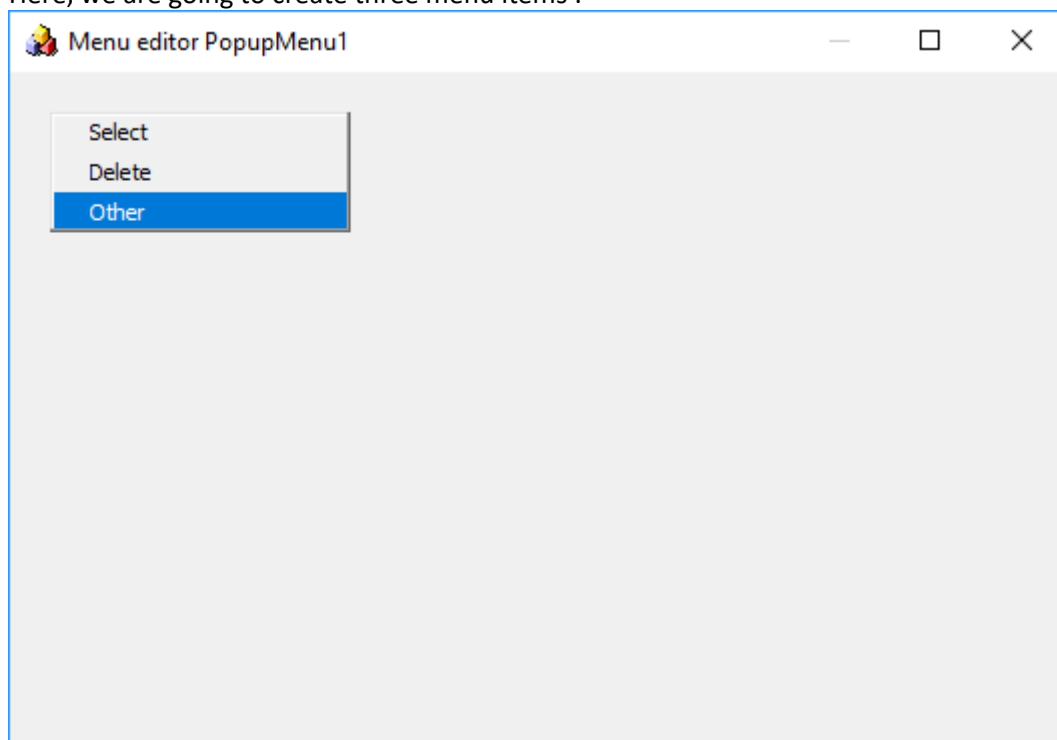
b) Add items

Double click on the CPopupMenu component

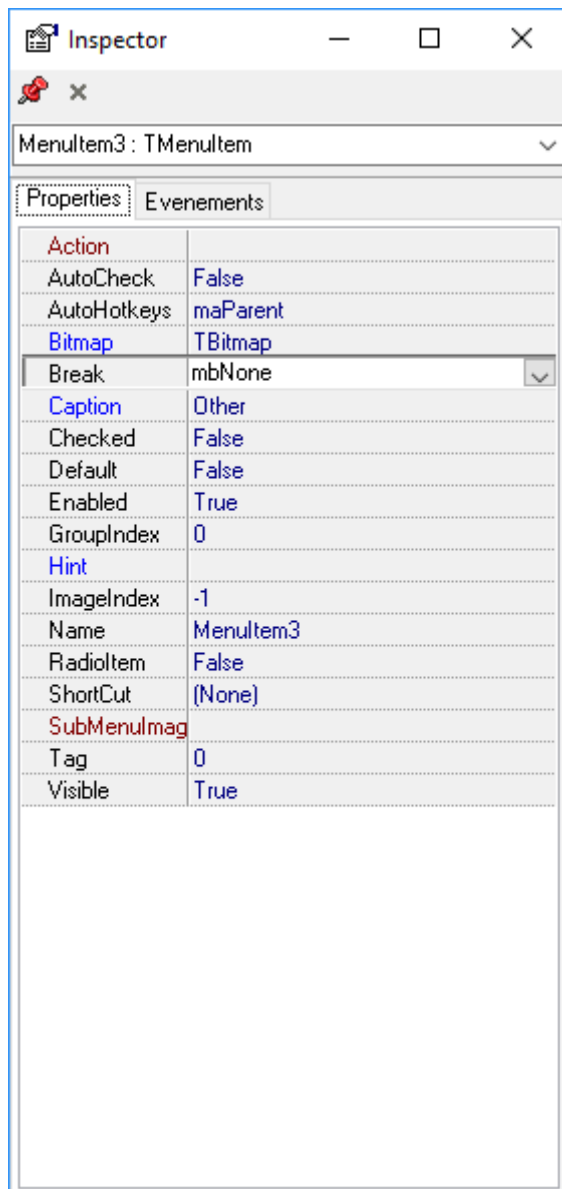
The menu editor open, right click in this editor, and select "New"



Here, we are going to create three menu items :



Properties such as displayed text (caption property) can be modified in property inspector



c) OnClick event

For each item, check onClick event, and enter the corresponding code :

```

*/EVENT DS3_FSFL_sf11Item2_OnClick
* -----*
* Description :
* -----*
D Parameters      ds          based(pevtnf)
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

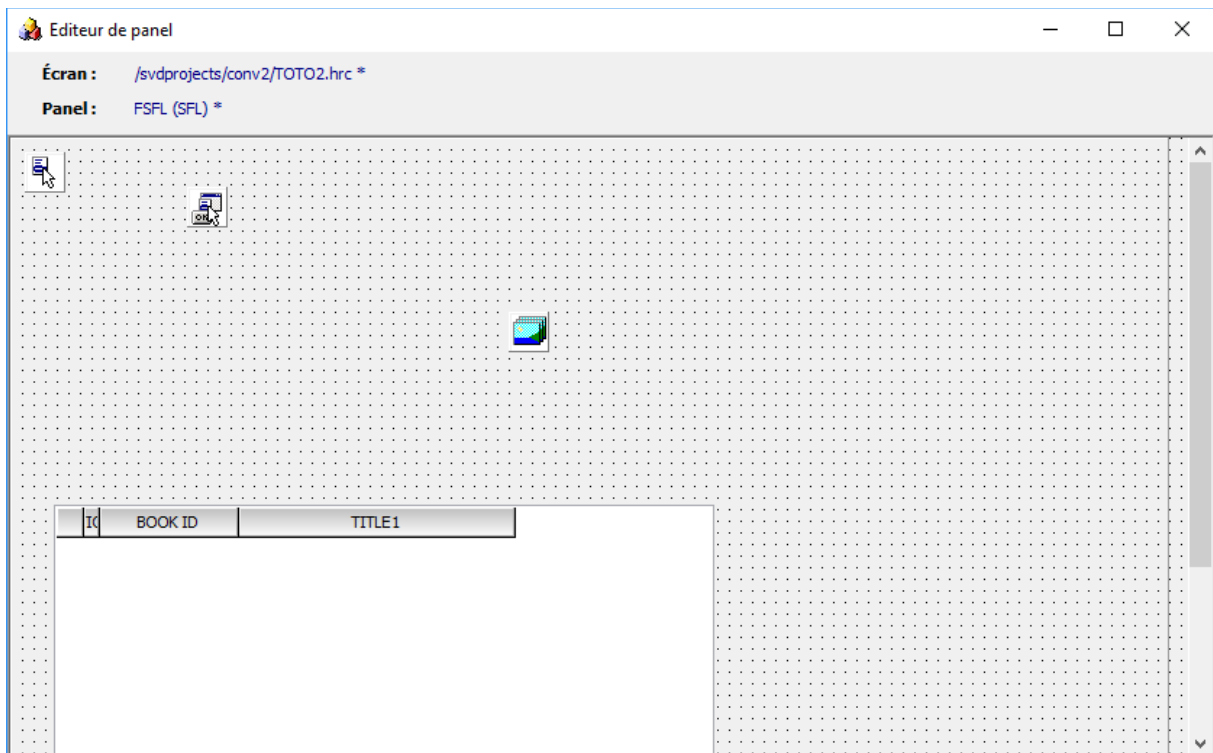
```

Code is identical to double click event, only option value is changed.

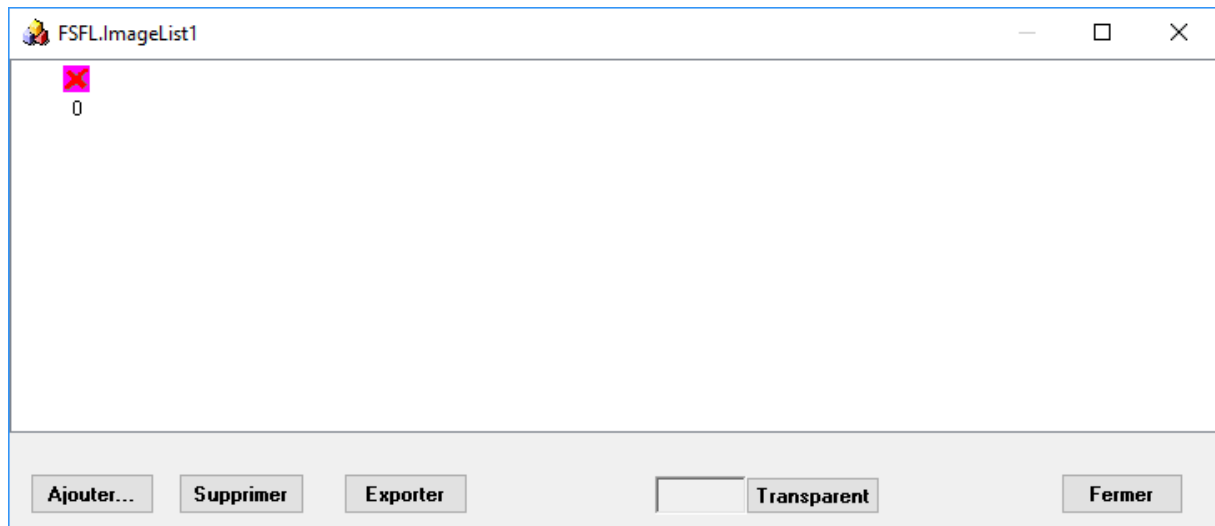
Note : If you checked “print” or “export” in the conversion properties form, a popup menu has been created and assigned to sfl. Add your menu items to this popup menu.

8. Adding images to menu items

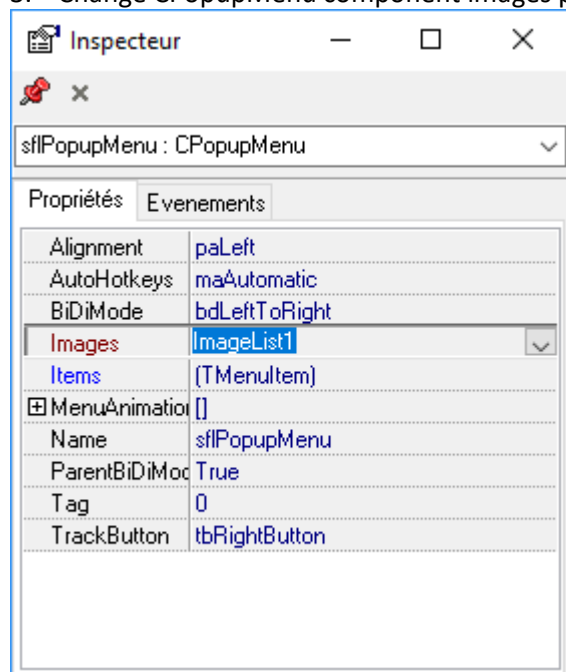
1. Add a CimageList component on the panel



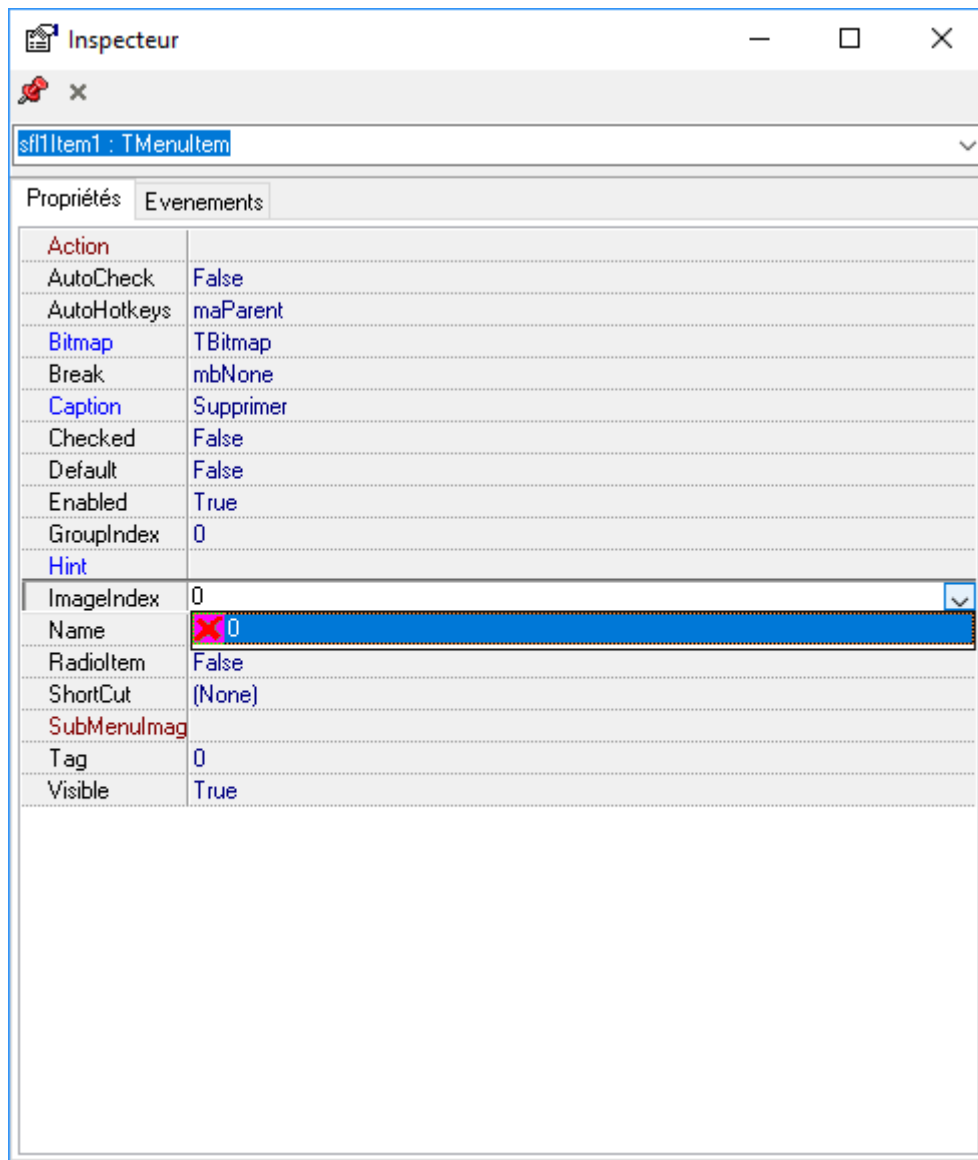
2. Add images in the CimageList component



3. Change CPopupMenu component images property



4. Change imageIndex properties of menu items



9. Multi row operations

If 5250 version handle an option on several rows, you can add an menu item that does the same. For exemple, if option 4 allows to delete several lines, change the options property of the CSFL component and add the goMultiselect option :

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	(OptionsGeneration)

Create a menu item as seen before, and enter the following code in onClick event :

```

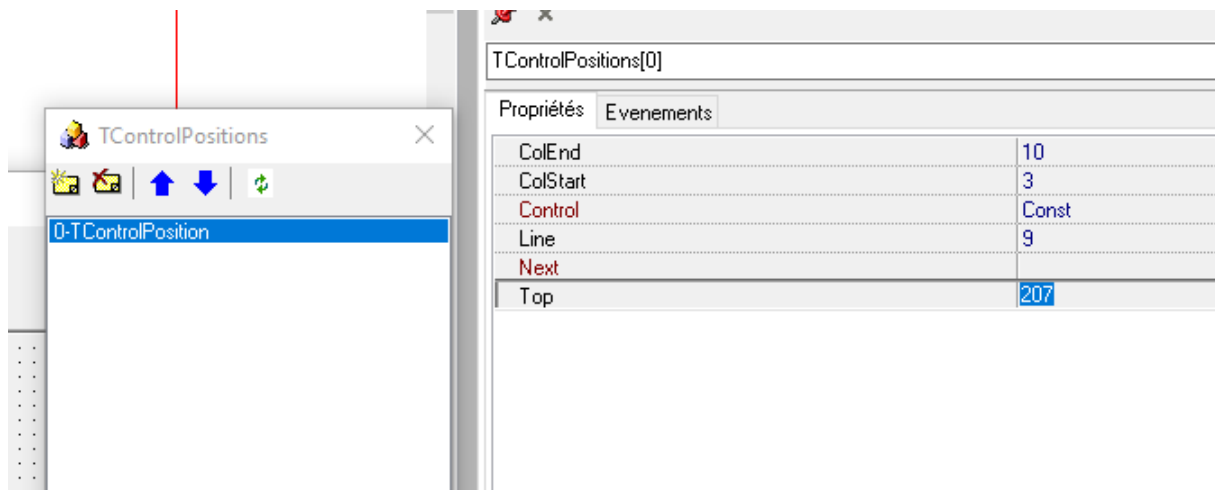
*/EVENT DS3_FSFL_miDelete_OnClick
* -----*
* Description :
* -----*
D Parameters      ds      based(pevtnf)
D Win              5u 0
D Evt              48
D sf1              s      *
D i                s      10u 0
sf1 = sdGetsf1(stateInfo.F1:'Ds3.format1.sf11':Selected_rows);
if sf1 <> *NULL;
  for i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

```

10. OVERLAY,CLRL

Handling key words OVERLAY or CLRL is not depending on components disposition but on disposition they had on 5250 screen. Informations are stored in the controlPositions property of the panel.

If you add a component (an image or a button for example) and you want that this component behave like the others with the keyword overlay and clrl, that means they are erased if a record overlay them, you have to add an item in the controlPositions property of the panel.



C. Modification in the copied program

Copied program can be modified. It is possible for example to replace a call to a system program by a silverdev program.

D. Modification in the handler and the copied program.

1. Sharing data

In order to let the handler program and the converted program communicate, it is possible to add to the instruction workstn handler('HDL0001') a second parameter :

workstn handler('HDL0001':USERAREA)

where userarea is a dataStructure of your choice.

The handler program access to this data structure by using the variable handlerParm.userArea.

Example

Source : LIB2/H/USERDATA

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

Source of the copied program :


```

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

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

before instruction write format0 :

userData.stuff1 = 'hello world';

```

In the Silverdev screen, on the FORMAT0 record, we add a label named label1.

Source of the handler program :

```

        /copy h,userData

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

        //inf writeformat, before the instruction

        //sdhWriteFmt (F1: 'FORMAT0')

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

```

The handlerParm variable is a entry parameter added automatically to the generated program.

The copied program handles the logic of the program and the handler handles the screen, but is not aware of the data in the copied program.

Using userArea parameter allows to palliate to this.

2. Passing procedure pointers

In order to keep principle of separation of goals.

The origin program accessing to data and handler handling the interface, we are going to see how to add a graphic in the handler and read the data in the origin program.

a) Origin program

In the origin program, we add the following data structure :

```

D UserArea      ds
D
D ptrRefreshGraphic...
D
D
D
inz(%paddr(refreshGrahpic))

```

Screen file Declaration :

```

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

```

The refreshGraphic function that will read data and fill the graphic is the following :

```

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;

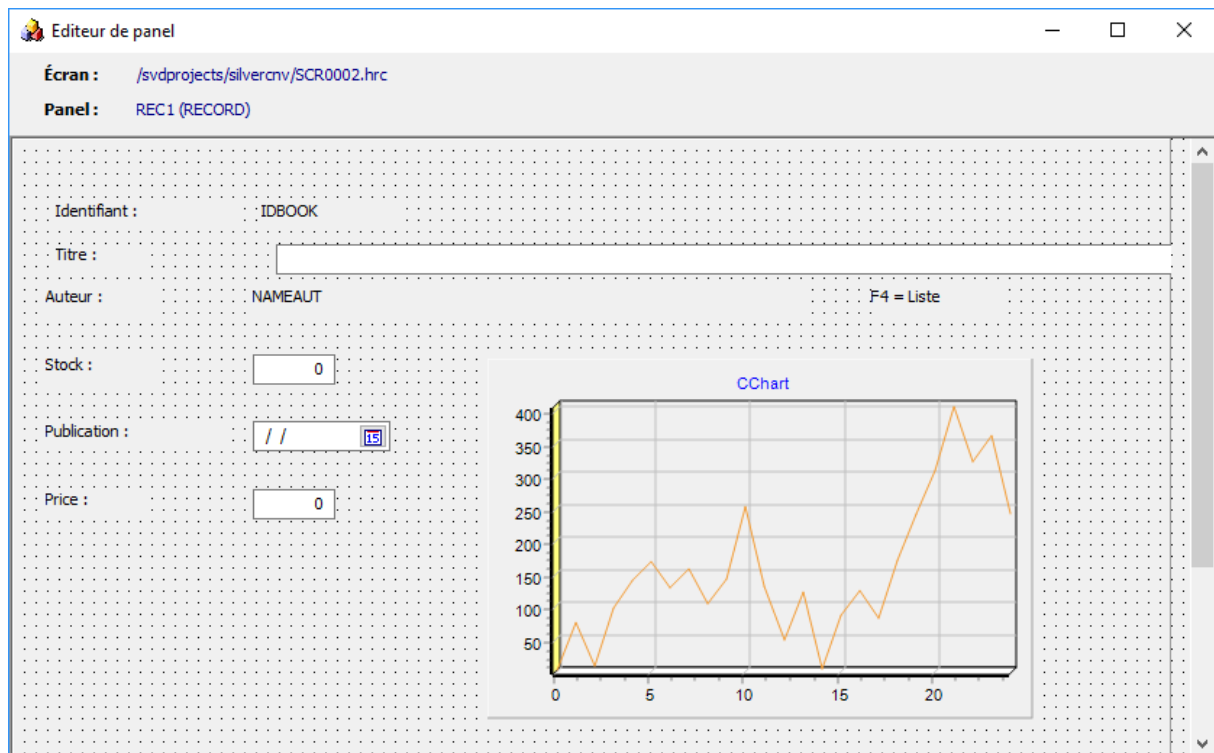
sdAddXY(win:component:date2.all:quantity:date1);
reade pData.idBook sddmorder1;
enddo;

/end-free
P refreshGrahpic...
P                                     E

```

b) Handler

In the handler, we add on a panel, a CChart component:



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
```

Then the function definition

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

Finally, in the writeFormat_Rec1 function, we add a call to the refreshGraphic function.

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

VII. Regeneration

A. Introduction

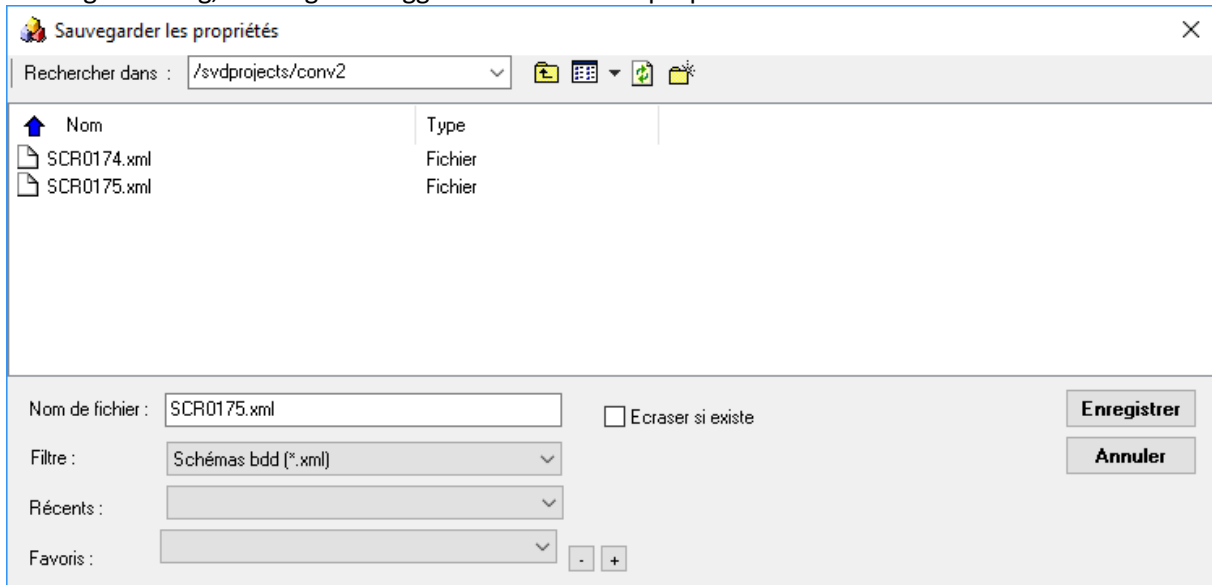
If the screen is modified, you have to re generate the handler.

In order to keep selected properties, this properties are stored in a xml file.
The default file name is the name of the screen with xml extension.

B. Save properties

A properties window allows to change the width of the generated window, function keys activated on closing cross, date fields, subfile popupmenu items...

While generating, a dialog box suggests to save these properties.



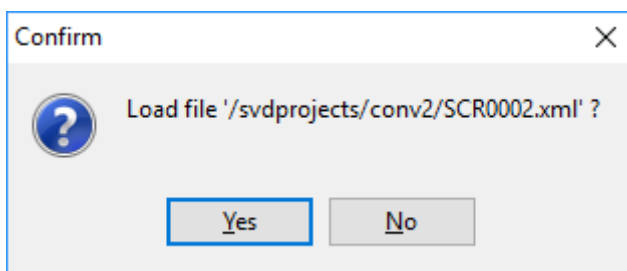
Properties are saved with xml format. You can then reload these properties if you want to re generate the handler.

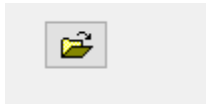
The suggested directory is the directory of the screen sources for the chosen context.

So, if you forgot to change a property when generating the handler, you can re do the generation while retrieving the properties you selected the first time.

C. Reloading properties

When opening the properties window, if a xml file with the name of the screen is found, designer suggests to load this file :





You can load it later, or you can chose another xml file by using the button

D. Regenerate menu

If the 5250 screen is modified, you have to re generate the handler.

In order to make easier handlers re generation, you can click on the context, and select regenerate.

Program	Description	Window	Window library	Display mode	Type	Screen source path	Display file	Display file library
SCR0001	Handler for SILVERDEMO/SCR0001	SCR0001	*LIBL	*MODAL	SVDHRPG	*DEFAULT	SCR0001	SILVERDEMO
SCR0002	Handler for SILVERDEMO/SCR0002	SCR0002	*LIBL	*MODAL	SVDHRPG	*DEFAULT	SCR0002	SILVERDEMO
SCR0003	Handler for SILVERDEMO/SCR0003	SCR0003	*LIBL	*MODAL	SVDHRPG	*DEFAULT	SCR0003	SILVERDEMO
SDDMCNV1	Handler for SILVERDEMO/SDDMCNV1	SDDMCNV1	*LIBL	*MODAL	SVDHRPG	*DEFAULT	SDDMCNV1	SILVERDEMO
SDDMCNV2	Handler for SILVERDEMO/SDDMCNV2	SDDMCNV2	*LIBL	*MODAL	SVDHRPG	*DEFAULT	SDDMCNV2	SILVERDEMO
SDDMCNV3	Handler for SILVERDEMO/SDDMCNV3	SDDMCNV3	*LIBL	*MODAL	SVDHRPG	*DEFAULT	SDDMCNV3	SILVERDEMO
SVDMBKS1	Handler for SILVERDEMO/SVDMBKS1	SVDMBKS1	*LIBL	*MODAL	SVDHRPG	*DEFAULT	SVDMBKS1	SILVERDEMO
SVDMBKS2	Handler for SILVERDEMO/SVDMBKS2	SVDMBKS2	*LIBL	*MODAL	SVDHRPG	*DEFAULT	SVDMBKS2	SILVERDEMO

- New
- Properties
- Source
- Delete
- Refresh F5
- New from...
- Copy to...
- Generate program
- Generate module
- Re generate handler
- Generate via Arcad
- Checkout (Arcad)

Designer deletes selected handlers and display batch generation form seen previously.

E. Save manual modifications

If you have manually modified the handler (screen or rpg), you have to update the conversion data file with the following menu :

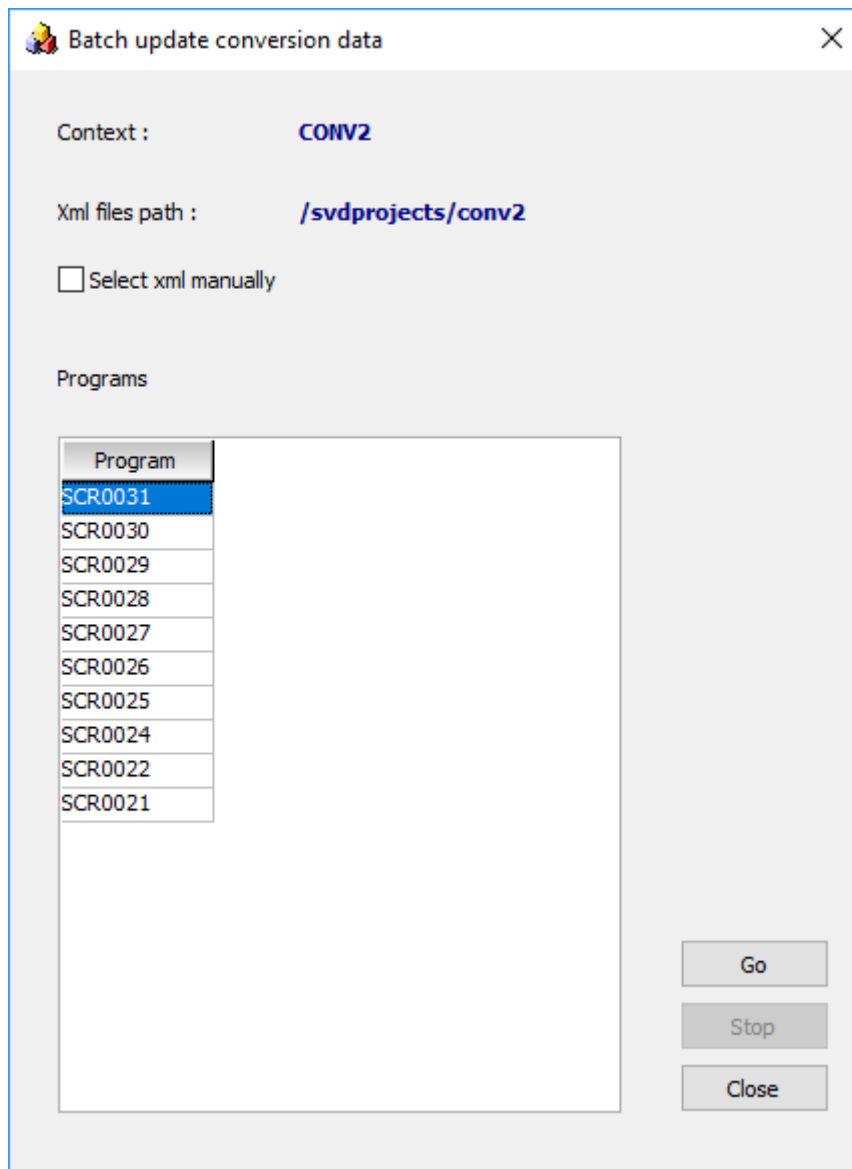
SCR0007	Handler pour CONV1/SCR0007	*PGM	*LIBL
SCR0007B	Handler for CONV1/SCR0007B	SCR0007B	*LIBL
SCR0008	Handler for CONV1/SCR0008	SCR0008	*LIBL
SCR0009			*LIBL
SCR0009B			*LIBL
SCR0010			*LIBL
SCR0011			*LIBL
SCR0012			*LIBL
SCR0013		F5	*LIBL
SCR0014			*LIBL
SCR0015			*LIBL
SCR0016			*LIBL
SCR0017	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

Not, for this operation, designer, needs screen and rpg to be open

A dialog box list programs that conersion data files will be updated.

If the checkBox « Select xml files manually » is checked, a dialog box is displayed to the user for him to select the xml file do update.

If this checkBox is not checked, the xml file with same name as the program in the path configured for the context is updated.



F. Removing components

It may be easier to remove components from a generated screen than check items in properties form.

It is possible that rpg code is linked to this removed components.

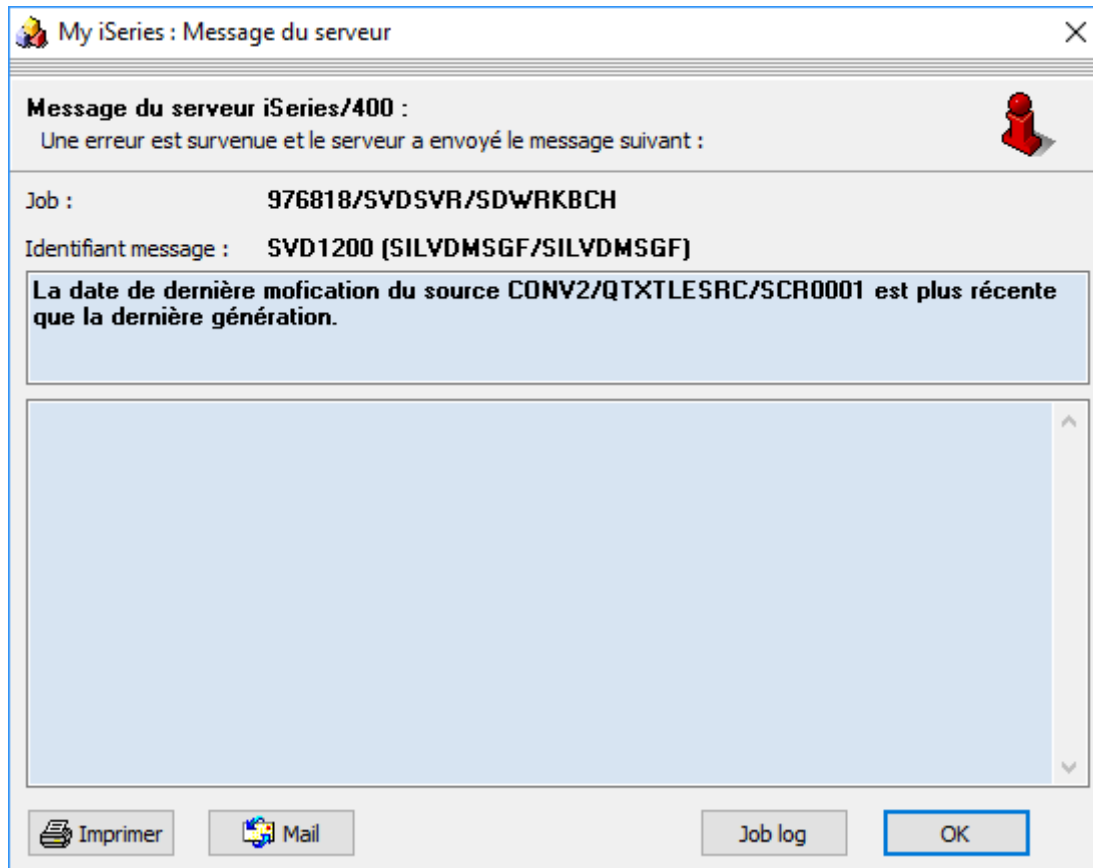
In this case, you have to save manual modifications, and re generate.

Save manual modifications will save fact that component does not exist anymore, and re generation will take it in account (component will not be generated , neither rpg code)

G. Protection

1. Without loading conversion data

When check box « Load conversion data » is not checked (or if the file is not found), designer checks that date of last generation is younger than date of screen source or qtxtlesrc member.



2. With loading conversion data

When check box « Load conversion data » is checked , designer checks that date of xml type data conversion file is younger than screen source and qtxtlesrc member.

If not, you have to update data conversion file. (see chapter « Save manual modifications »)

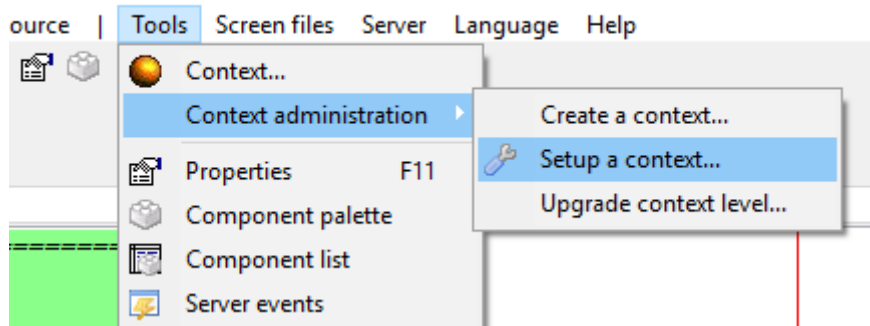
3. Skip protection

To force regeneration, check checkbox "Re generate even if modified"

VIII. Set default values

You can configure default values suggested by properties form.

Proceed as follow :



Context setup

General
Conversion

Default values for generation

Window width	800
Window height	600
Marge	20

OnClose function key

F3

F12

Conversions

Fields 6,0	
Fields 7,0	
Fields 8,0	

SFL

☒ Add print sfl

☒ Add export sfl

☒ Alternate sfl lines color

Odd lines :

Even lines :

Header height
0

☒ Left column

OK

Cancel

A. OnClose function key

For the function key activated on onClose function key, the first value will be used in priority, then the second one, then the third.

B. Conversions

This settings allows to select automatically a conversion type for numeric fields.

C. SFL

This setting allows to give default values for CSFL component created.

IX. Constraints

Since RPGOA technology is used, minimum operating system is V6R1.

An operating system V7R1 is better, it allows to user teraspace storage model. Some handlers will not compile without it .

Only RPG programs can be converted.

You need RPG program sources.

Origin program must not call 5250 interfaces.

- All called programs must be converted
- You cannot call system programs, like wrksplf
- command line
- origin rpg program must not generate 5250 flow.

Only extern screens are handled.

When you compile the copied program and when you run it, 5250 screen must be in library list.

When a program does not match these conditions, you can modify the copied program or re write entirely a classic silverdev program thanks to compatibility between classic silverdev programs and converted programs.

X. RPG 3 programs

Handler instruction is available only in RPGLE. To convert a program RPG in RPGLE, use the CVTRPGSRC command.

Note :

When CVTRPGSRC ends with following message :

RNS9378 Log file QRNCVTLG in library *LIBL not found..

You can create this file with the CRTDUPOBJ command (and not CPYF as mentionned in message help) by copying the file QRPGL/QARNCVTLG

XI. Replacement solutions

A. Calling a system program

System programs can be re written with silverdev, using IBM apis.

This programs can be called in many programs, this work can be reused.

Experia already provides some programs of this type and the number os system programs provided by Experia is going to increase with time.

B. Command line

If a command line is displayed in the origin program, it is possible to add a CCmdDialog component.

Example :

C. Dsply

A call to dsply can be replaced by the function sdShowMessage.

You have to add following instructions to the program :

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

D. Deleting

if a program cannot be re written because it is to linked to 5250, or if you want to report its development in the future, you can deactivate the key function that calls this program. (Si chapter Deactivation of function keys)

A. RECVF

For one of our client, we found the following code :

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

So a waiting window was displayed for 3 seconds

We replaced by this :

```

H DFTACTGRP (*NO)
FWD001FM  CF  E          WORKSTN
F                                     handler('WD001FM')

/free
  *inlr = *on;
  exfmt WD001WW;
/end-free

```

On the silverdev screen, we added a timer component set to three seconds, and in the onTimer event :

```

*/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

```

XII. Reference fields

If a screen field is marked with the keyword REFFLD, the library of the file used when the dspf was compiled is indicated in the DSPF.

The conversion tool will search for the file in this library, and if it does not find it, in the library list.

XIII. Licences

To use the conversion tool, you need a development licence with the conversion option.

If the conversion option is not active or outdated, the menu item "Screen file/generate a handler" is deactivated.

To execute converted programs, the runtime licence is enough.

So, even when the conversion licence is outdated, converted programs work.

You still can modify converted programs manually when conversion licence is outdated.

XIV. Several screens in a program

If a program call several screens, you have to create a handler for each screen.

Example :

<pre>H DFTACTGRP(*NO) FSCR0167 Cf e FSCR0168 Cf e /free *inlr = *on; dow not *in03; exfmt RECORD1; if *in04; exfmt RECORD2; endif; enddo; /end-free</pre>	<pre>workstn handler('HDL0167') workstn handler('HDL0168')</pre>
--	--

XV. Habitual mistakes

CPF4103	Handler keyword has not been added and the program searches for the origin (dspf)screen It is also possible the copied program has not been compiled, and the origin program is called instead of the copied program
CPF4103	A program in the program chain has not been converted, and the origin program has been called.
MCH3401	Handler program has not been compiled
SVD4130	Copied program has been compiled with a different version of 5250 screen. Solution : Re compilel program
SVD4131	Handler program has been compile with a different version of 5250 screen. Solution : Re generate handler program
MCH3402	5250 screen has been deleted while program whas running. Problem happens while closing

	the silverdev window.
User	If in 5250 RPG programs you use the field at offset 254 of sds, with Silverdev, this field will give you the profile that started silverdev server. You have to replace by offset 358.

XVI. Full example

A. Présentation

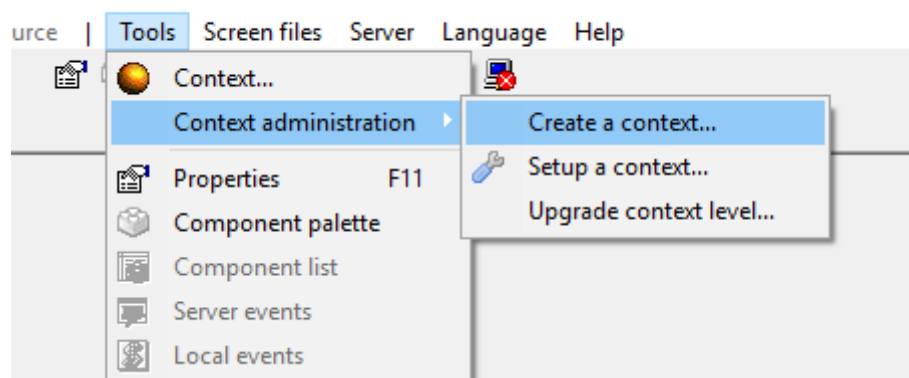
In Silverdemo library, there are 5250 programs.
 These programs are PGM0001 to PGM0003
 Screens are SCR0001 to SCR0003.

The first screen displays a book list, option 2 allows to display book form and f4 on book form allows to select an author.

Originals programs , data base files and 5250 screens are in SILVERDEMO library.

B. Context creation

We start by creating a library SILVERCNV and a SILVERCNV context.



*LIBL/CRTSDCTX Create SilverDev context

Prompt Text

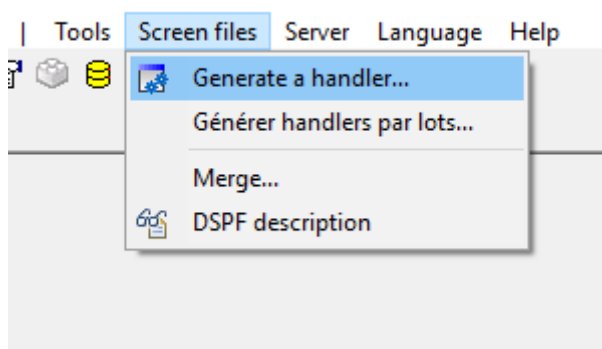
CRTLIB
TOLIB

TOLIB
Create context into library SILVERCNV Name

CRTLIB
Create library YES *NO, *YES

Submit Execute Cancel

C. Handlers generation



Handler generation

5250 screen

Library : SILVERDEMO ...

Name : SCR0001 ...

Silverdev program

Context : SILVERCNV ...

Program : SCR0001

Screen : *PGM

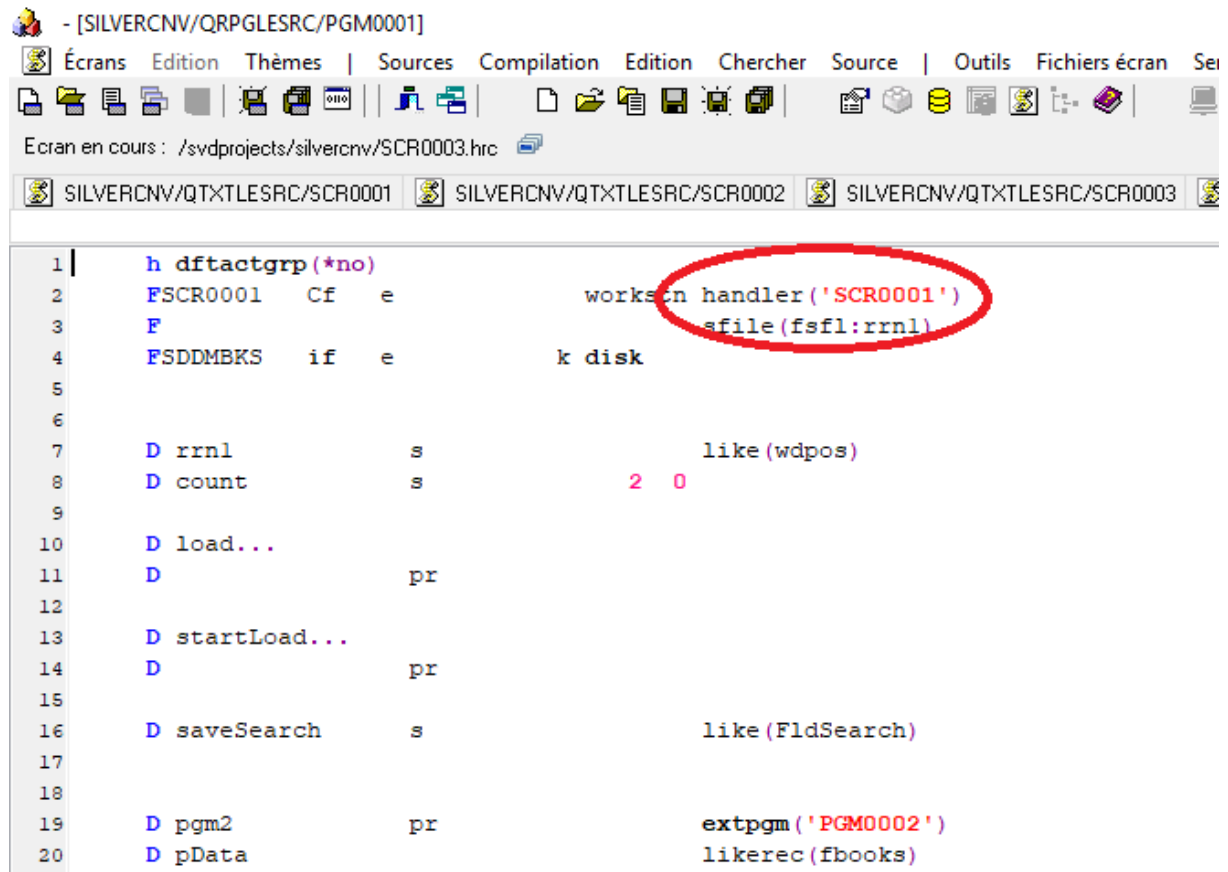
Type : SVDHRPG

Description : Handler for SILVERDEMO/SCR0001

OK Cancel

D. Copy origin programs

We copy origin programs in silvercnv and we add instruction handler in screen declaration.



```
- [SILVERCNV/QRPGLESRC/PGM0001]
Écrans  Edition  Thèmes | Sources  Compilation  Edition  Chercher  Source | Outils  Fichiers écran  Sei
Ecran en cours : /svdprojects/silvercnv/SCR0003.hrc
SILVERCNV/QTXTLESRC/SCR0001  SILVERCNV/QTXTLESRC/SCR0002  SILVERCNV/QTXTLESRC/SCR0003

1 | h dftactgrp(*no)
2 | FSCR0001 Cf e workspace handler('SCR0001')
3 | F sfile(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. Starting converted program

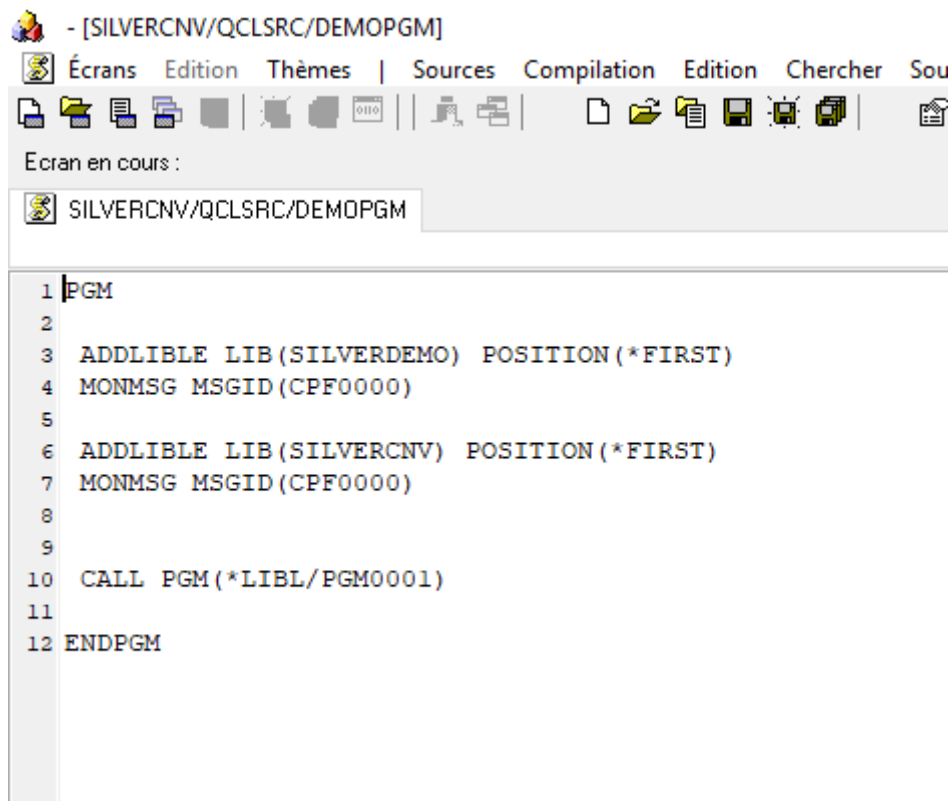
In Silvercnv/qclsrc, we write a starting program.

We add SILVERDEMO and SILVERCNV in library list.

SILVERCNV must be before SILVERDEMO for converted programs been used and not origin programs.

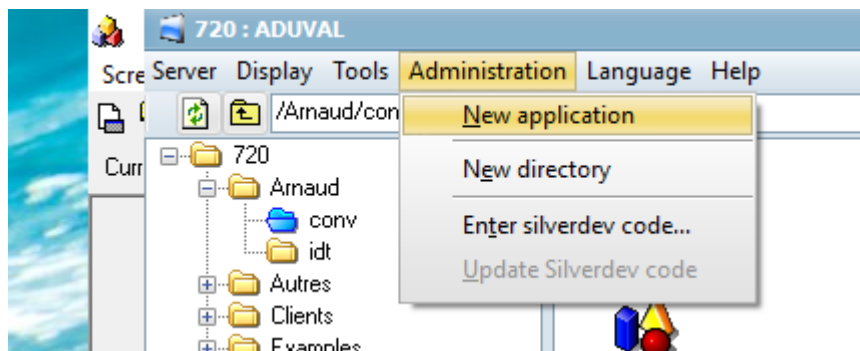
SILVERDEMO must ben in library list because 5250 screens must be in library list.

Also, data base files are in SILVERDEMO.



We add an icon in MyDesk :

(Profile used for connection must ben in file SILVERDEV/PSVDADM for the administration menu to be visible)



New [X]

Title :
Demo program

App file name :
DemoPgm

Command :
call silvercnv/demopgm ...

Description :

☐ Unique run

Icône : [] ... []

Intern icon : [] ... []

Rank : -1

[OK] [Cancel]

Start the 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

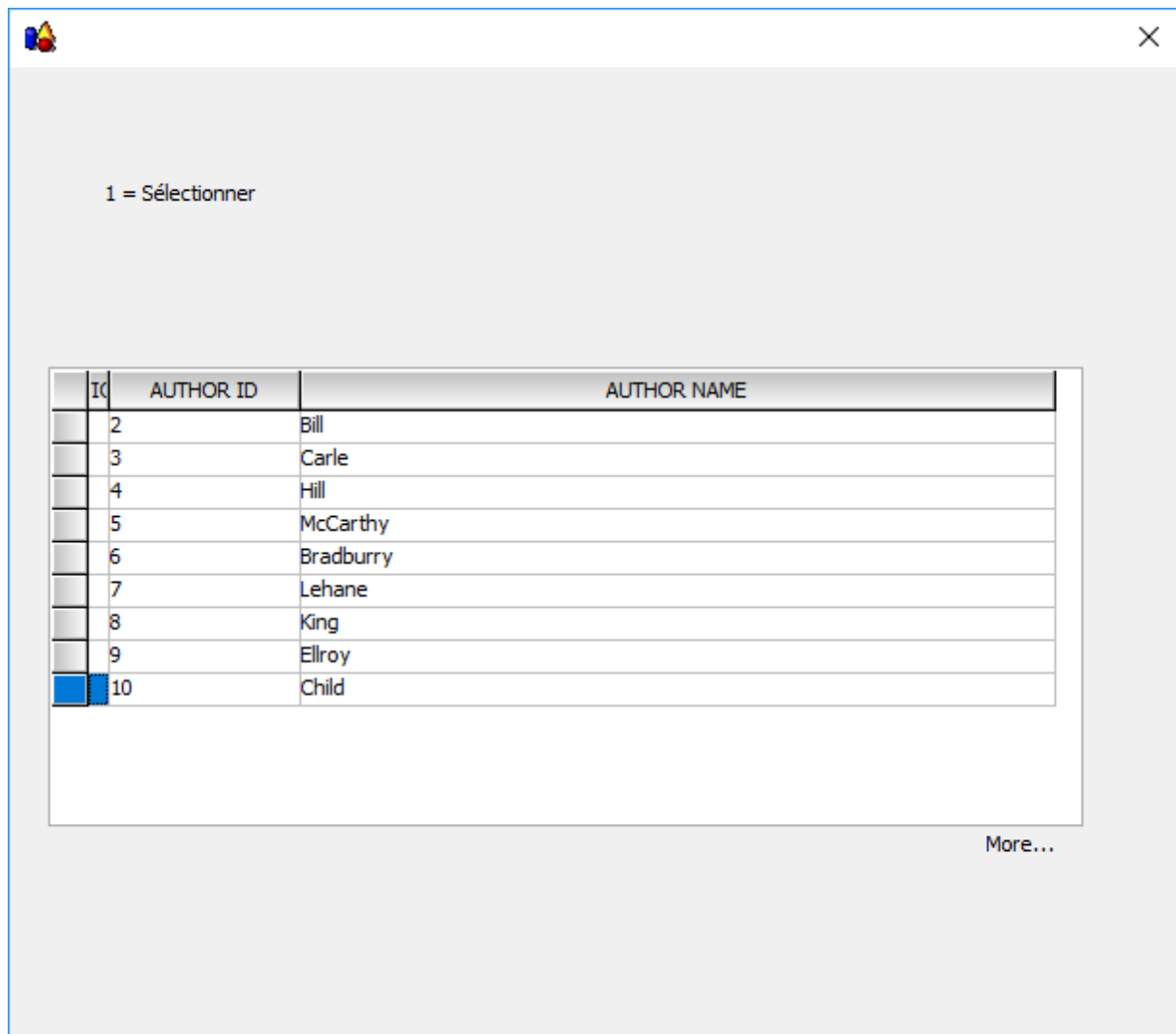
Titre :

Auteur : Carle F4 = Liste

Stock :

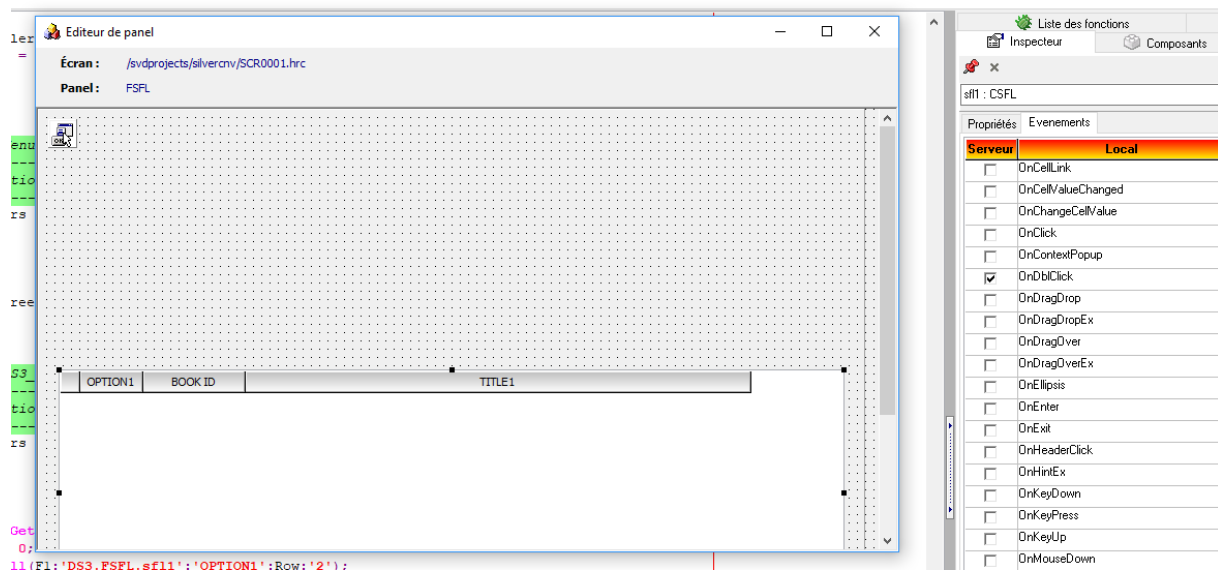
Publication :

Price :



F. Modifications in generated programs

1. Adding a double click on subfiles :



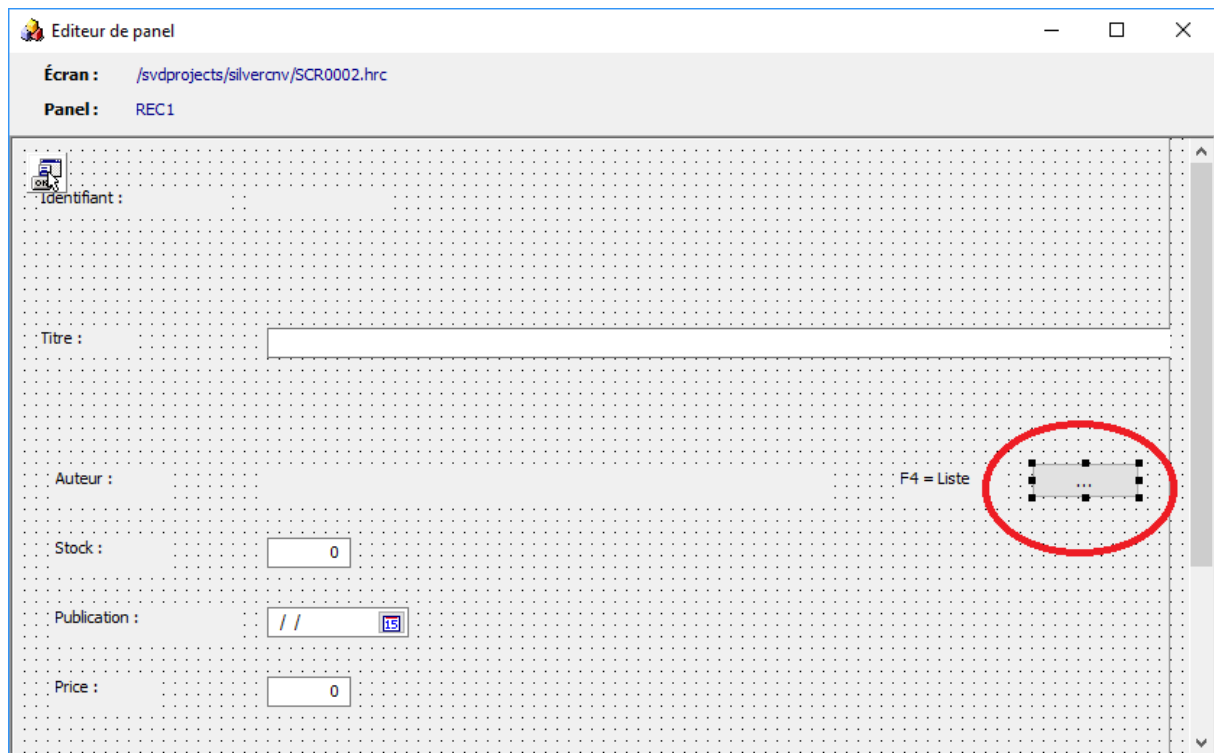
```

*/EVENT DS3_FSFL_sf11_OnDbClick
* -----
* Description :
* -----
D Parameters      ds      based(pevtnf)
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. Adding a button in pgm2

In pgm2, we add a button that will open the same window as F4 function key.



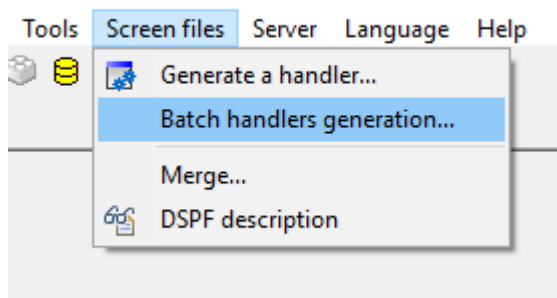
```

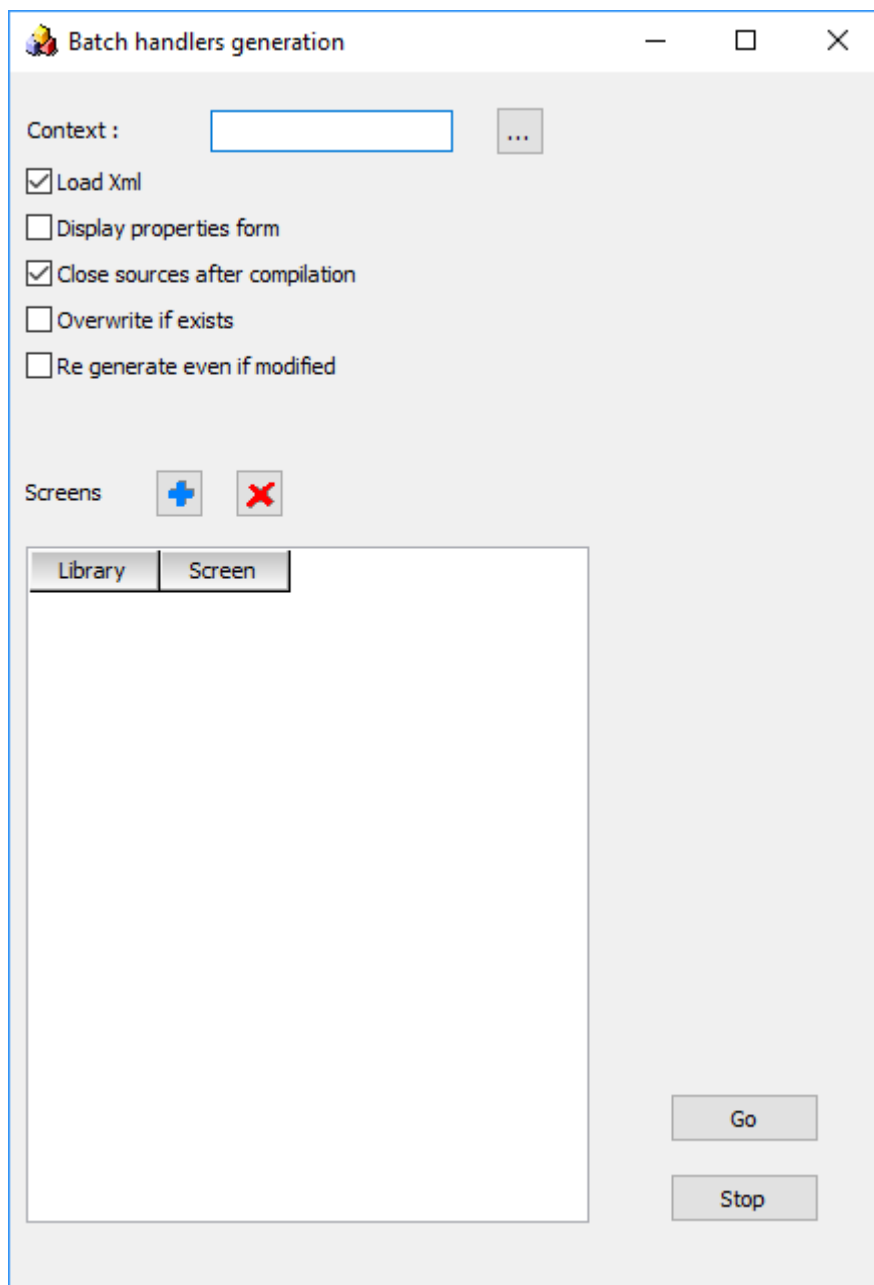
*/EVENT DS3_REC1_btnSearchAuthor_OnClick
* -----*
* Description :
* -----*
D Parameters    ds          based(pevtnf)
D Win           5u 0
D Evt           48
/free
  REC1_IN.FLD = 'NAMEAUT';
  DS3_REC1_CF04_OnExecute();
/end-free

```

XVII. Multiple generations

To generate several handlers, click on "Screen files/Batch handlers generation..."





XVIII. KeyWords

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 generated before sdExFmt.
2	Alias	Alias is not supported by rpg. This would have no effect anyway on the handler
3	Althelp	Function help called
4	Altname	Altname is not supported by rpg. This would have no effect anyway.
5	Altpagedwn	Event Rollup_onExecute called

6	AltPageUp	Event RollDown_OnExecute called
7	Alwgph	GDDM not supported -> keyword not supported
8	Alwrol	Not supported in rpg
9	Assume	Records saved in variable KeepPanels are displayed when screen file is opened
10	Auto	Same as check
11	Blanks	Test sdGetText = *blanks
12	Blink	Nothing
13	Blkfold	Nothing to do because Silverdev does not fold, there are scrollbars
14	Cann	Event FNN_OnExecute is changed
15	Cfnn	Event FNN_OnExecute is changed
16	Change	Indicator is put to *on
17	Chcaccel	Radiobutton caption is modified
18	Chcavail	Not Important
19	Chcctl	Component state changed
20	Chcslt	Nothing
21	Chcunavail	Nothing
22	Check	CHECK(ME) : test field blank CHECK(LC) charcase property CHECK(RB) right aligned and evalr Check(AB) If comp, range or values, blank added CHECK(MF) test field full filled CHECK(M10) call to sdCheck('M10'... CHECK(M11) call to sdhCheck('M11'... CHECK(M10F) onchange, call to sdhCheck('M10'... CHECK(M11F) onchange, call to sdhCheck('M11'... CHECK(VN) call to sdhCheck('VN'... CHECK(VNE) call to sdhCheck('VNE'... CHECK(RZ) displayFormat filled with blanks CHECK(ER) not handled CHECK(FE) not handled CHECK(RL) not handled CHECK(RLTB) not handled
23	Chginpdft	Nothing to do
24	Chkmsgid	Call to sdRtvMsg
25	Choice	For sngchcfld, creation of a radiobutton
26	Chrid	No ccsid for client part. All fields are displayed in rpg programm ccsid
27	Clear	Event Clear_onExecute
28	Clrl	Some controls are removed
29	Cmp	Controlled at validation
30	Cntfld	Generation of a normal field, taken in account for overlay.
31	Color	Font.color modified. (if green -> clWindowText)
32	Comp	Idem cmp
33	Csrinponly	TabStop put to false depending on CSRINPONLY
34	Csrloc	Call to sdGetControlAt
35	Date	%char(%Date)
36	Datfmt	Taken in account for buffers declaration

37	Datsep	Taken in account for buffers declaration
38	Dft	Value put by default if field in output and first write
39	Dftval	Value put by default if field in output and first write or if field in input
40	Dltchk	Nothing to do, informations not added in screen file,so not read by designer
41	Dltedt	Nothing to do, informations not added in screen file,so not read by designer
42	dspatr	Modification of font, color, readonly, visilble properties, cursor position
43	Dspmod	Correct panel is displayed
44	Dsprl	Not handled
45	Dspsiz	For each record, a panel is create by dspSize
46	Dup	Not handled
47	Edtcde	%editc generated
48	Edtmsk	Corresponding editMask property generated.
49	Edtwrd	if zero decimals ,creation of a maskEdit and adaption of the editmask property if decimals, createion of an editnum, generation of the displayFormat property
50	Entfldatr	Nothing to do
51	Erase	Generation of sdhHideFmt
52	Eraseinp	Fields alread displayed and not protected put to blank. Si eraseinp(*mdton), seulement les champs avec mdt on
53	Errmsg	Error displayed at exectuion.
54	ErrMsgld	Idem ErrMsg
55	Errsfl	Error messages are always put in a sfl in silverdev. (could change in the future)
56	Fldcsrprg	Newt property added in TControlPosition
57	Fltfixdec	Works like that. A component dedicated to floats could be developped
58	Fltpcn	Works
59	Frcdta	Frcdta property on panels.
60	Getretain	Call to sdhUnlock
61	Help	Help is displayed
62	Hlpara	Help is displayed depending on the cursor
63	Hlpbdy	Code in function Help takes account of this
64	Hlpclr	Help display is conditionned in program
65	Hlpcmdkey	End of red on main screen added.
66	Hlpdoc	Not handled
67	Hlpexcl	Ok
68	Hlpfull	Help is always displayed in same window.
69	Hlpid	This keyword impacts other keywords that are handled
70	Hlppnlgrp	Help is displayed
71	Hlprcd	Help is displayed
72	Hlprtn	Help is not displayed, response indicator is put to *on
73	Hlpschidx	Not handled No api to read search index
74	Hlpseq	On rollout_OnExecute, correct help record is displayed
75	Hlptitle	Window title is changed
76	Home	Call to sdhHome
77	Html	No need

78	Indara	Generated code is different , indicators are passe to the origin program via an array and not via in/out buffers
79	Indtxt	Used in the designer to help the developper
80	Invite	MAXDEV not compatible with RPGOA -> invite not compatible
81	Inzinp	Save area is modified
82	Inzrcd	
83	Keep	Displayed records are save in the variable keepPanels from sdsrvpgm service program
84	Lock	Call to sdhUnlock..
85	Loginp	Call to qmhsndpm to add input field values in job log
86	Logout	Call to qmhsndpm
87	Lower	Charcase property
88	Mapval	Values changed
89	Mdtoff	Call to sdhHandleMdt
90	MLtchcfld	Creation of several checkbox
91	Mnubar	Creation of a mainmenu
92	Mnubarchc	Creation of a menuitem
93	Mnubardsp	MainMenu is assigned
94	Mnubarsep	Nothing
95	Mnubarsw	Associated function key is not activated
96	Mnucnl	PullDown window is closed (sdClossPullDown...
97	Moubtn	Code generated on events onMouseDown and onMouseUp
98	Msgalarm	Call to sdBeep
99	Msgcon	Caption modified
100	Msgid	Changed at runtime with sdRtvMsg
101	Msgloc	No need , messages are displayed in a separate window
102	Noccsid	Not handled
103	Openprt	Prtf file not closed
104	Overlay	Overlay property on panels is changed at runtime
105	Ovratr	Components properties are modified
106	Ovrda	Fields are update depending on putovr, ovrda and first display
107	Pagedown /pageup	Idem rollup
108	PageUp	Idem PageDown
109	Passrcd	
110	Print	Screen sent in spool, or response indicateor put to *on
111	Protect	Call to sdhProtect
112	Pshbtnchc	Adding a button
113	Pshbtnfld	Adding buttons
114	Pulldown	Displayd on menuitem click
115	Putovr	Components properties modified (data and attributes) if putovr is active (and ovrda or ovratr)
116	Putretain	Output fields : Value returned only if not putretain
117	Range	Checked at validation
118	Ref	Impact other keywords
119	Reffld	Field is searched in database, properties are used
120	Retkey/retCmdKey	RetKey and retCmdKey properties on panels

121	Retlcksts	Call to sdhUnlock...
122	Rmvwdw	In Silverdev, windows are always removed
123	Rollup	Event Rollup_OnExecute
124	RollDown	Event RollDown_OnExecute
125	Rtncsrloc	Call to sdGetCtrl to know selected field
126	Rtndta	A similar mecanism is instored
127	Setof	Nothing to do, works by itself
128	Setoff	Idem setof
129	Sfl	Component CSFL type
130	Sflhcctl	Ok
131	Sfldlt	
132	Sflcsrprg	TabulationType property on the column
133	Sflcsrrrn	Returns row number of the sfl or 0 if not focused
134	Sflctl	Taken in account in generation
135	Sflclr	Call to sdClear on sfl
136	Sfldrop	Columns are all displayed on same line. Columns are hidden/displayed on function key. Columns are first hidden
137	Sfldsp	sdhWrite generated for sfl record
138	Sfldspctl	DisplayFields property modified
139	Sflend	A label or an image are added. vertical scrollbar is present whatever the value of *srcbar
140	sflenter	Last row is selected
141	Sflfold	Columns are all displayed on same line. Columns are hidden/displayed on function key. Columns are first displayed
142	Sflinz	Empty rows are added
143	Sflin	Subfile is displayed in lines, columns number is same as 5250 version
144	Sflmltchc	Adding a sfl with two columns. First column of type csBoolean
145	Sflmode	Always returns 0
146	Sflmsg	Message is displayed
147	SflMsgld	Idem SflMsg
148	Sflmsgkey	CSFL. values are read in program message queue
149	Sffmsgrcd	CSFL with values read in program message queue
150	Sflnxtchg	sdReinit is not called in update function if sflNxtChg
151	Sflpag	No impact on screen
152	Sflpgmq	CSFL, values read in program message queue
153	Sflrcdnbr	Call to sdGotoCell
154	Sflrna	sdReinit called if sflrna and sflinz
155	Sflrolval	pageHeight property
156	Sflrttsel	SFLRTNSEL property
157	Sflscroll	Call to sdGetInt on topRow property
158	Sflsiz	No impact on screen
159	Sflsngchc	CSFL with two columns, first column is of type csRadio
160	Slno	Slno property on panels, components are moved
161	Sngchcfld	CRadioGroup component
162	Sysname	Call to sdGetSysName
163	Text	No impact
164	time	%char(%Time)
165	Timfmt	Taken in account in buffers declaration
166	Timsep	Taken in account in buffers declaration
167	Unlock	Call to sdhUnlock

168	User	Call to sdGetCurUser
169	Usrdfn	Not handled
170	Ussrdspmgt	Window becomes transparent by modifying its properties transparentColor and transparentColorValue
171	Usrrstdsp	
172	Valnum	I valnum, minvalue is 0
173	Values	CCombobox component
174	Vldcmdkey	Indicator set to *on in onexecute action event
175	Wdwborder	Nothing to do
176	Wdwtitle	Window title modified
177	Window	New window in CsvdDspf, displayed modal
178	Wrdwrap	No impact, because field on one row