LeverX International, LLC
7A Lopatina str. 3 floor, Minsk, Belarus
Tel +375 17 232-20-01 http://www.leverx.by

Lection 7 Workbook

Controls, ALV and Print

Contents

LeverX International, LLC
7A Lopatina str. 3 floor, Minsk, Belarus
Tel +375 17 232-20-01 http://www.leverx.by

ALV control

Introduction

Here is the definition for ALV from SAP Help:
"The ALV Grid control is a flexible tool for displaying lists. The tool provides common list operations as generic functions and can be enhanced by self-defined options."
The ALV Grid control is used to build non-hierarchical, interactive, and modern-design lists. As a control, it is a component that is installed on the local PC.
The ALV Grid control provides typical list functions as sorting, filtering, summing, etc., while also gives the opportunity to develop user functions where needed. It presents numerous interfaces like Excel Inplace and Crystal Reports.

How to create simple ALV table - CL_SALV_TABLE

The main class used to create the simple 2D table is the class CL_SALV_TABLE. Create a reference variable for this class. Create an internal table and fill this internal table with data as shown below:

```
REPORT  zlx_cw7.

DATA: gt_lxcars TYPE TABLE OF zlxcars. *Internl table
DATA: gr_table TYPE REF TO cl_salv_table. *ALV object


START-OF-SELECTION.

SELECT * INTO TABLE gt_lxcars FROM zlxcars.
```

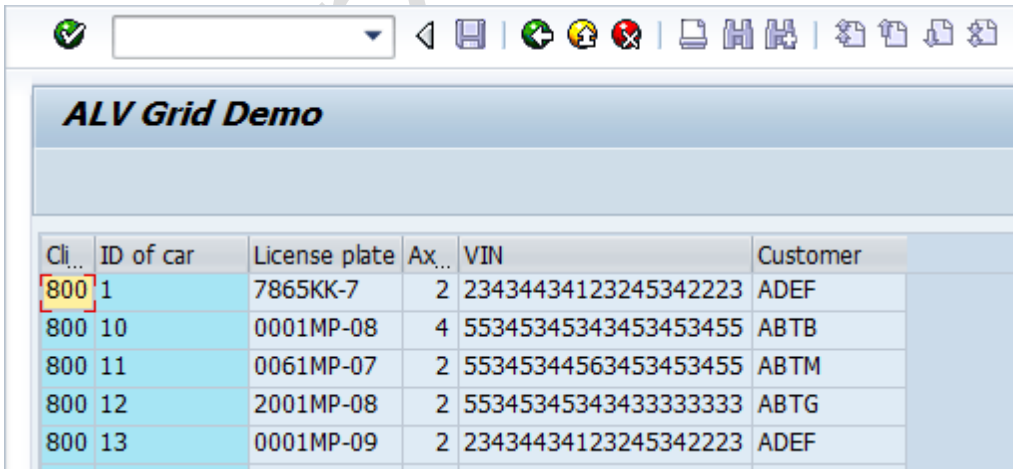
Next we need to create the ALV object for the 2D table. The FACTORY method allows you to create the ALV object in 3 ways. You can create the ALV Grid, as a classical list display, as a full screen grid, and finally embedded into a screen container. For this example, we will be work with the full screen grid. Create the call to the FACTORY method. We are importing the object reference into GR_TABLE and passing the internal table ZLXCZRS.

```
cl_salv_table=>factory( IMPORTING r_salv_table = gr_table CHANGING t_table = gt_lxcars ).
```

Next we need to display the grid, for this we use the DISPLAY method. Simply call it.

```
gr_table->display( ).
```

Result:



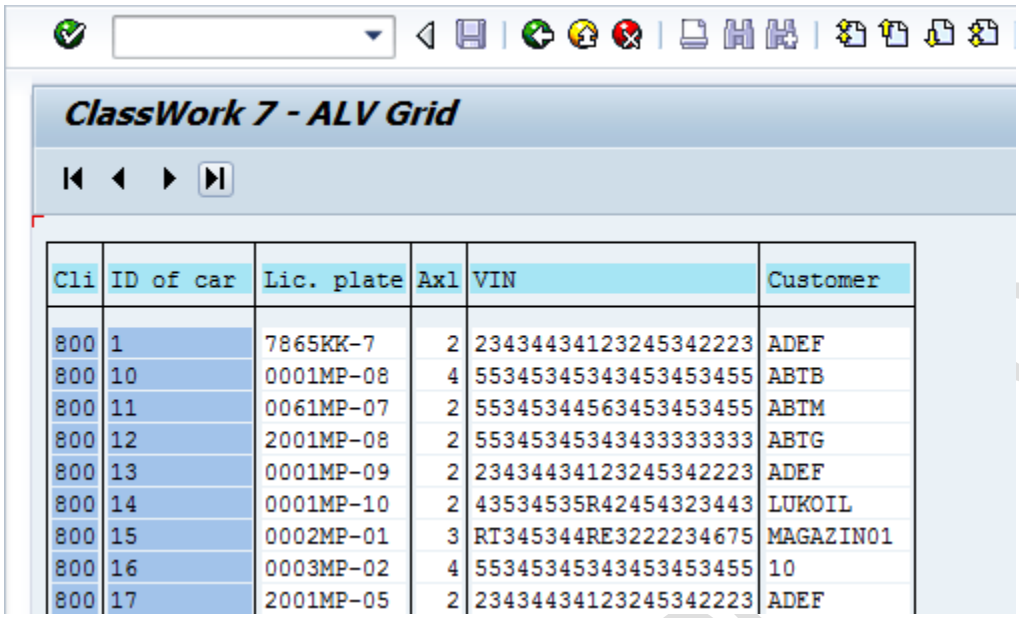
Cli...	ID of car	License plate	Ax...	VIN	Customer
800	1	7865KK-7	2	23434434123245342223	ADEF
800	10	0001MP-08	4	55345345343453453455	ABTB
800	11	0061MP-07	2	55345344563453453455	ABTM
800	12	2001MP-08	2	55345345343433333333	ABTG
800	13	0001MP-09	2	23434434123245342223	ADEF

LeverX International, LLC
7A Lopatina str. 3 floor, Minsk, Belarus
Tel +375 17 232-20-01 http://www.leverx.by

You can draw ALV as a list. For that you just need to add the LIST_DISPLAY parameter:

```
cl_salv_table=>factory( EXPORTING list_display = abap_true
                        IMPORTING r_salv_table = gr_table
                        CHANGING t_table = gt_lxcars ).
```

Result:



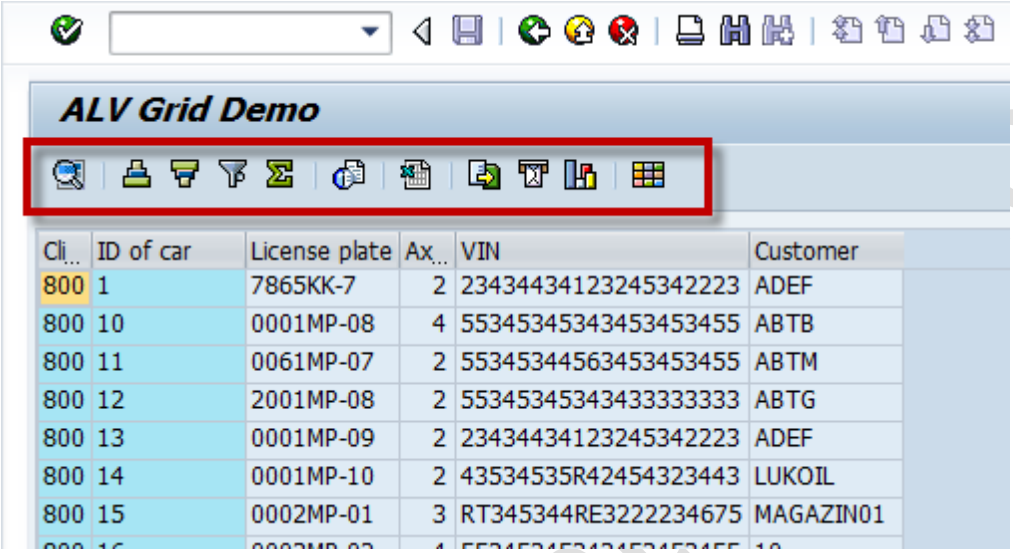
Cli	ID of car	Lic. plate	Axl	VIN	Customer
800	1	7865KK-7	2	23434434123245342223	ADEF
800	10	0001MP-08	4	55345345343453453455	ABTB
800	11	0061MP-07	2	55345344563453453455	ABTM
800	12	2001MP-08	2	55345345343433333333	ABTG
800	13	0001MP-09	2	23434434123245342223	ADEF
800	14	0001MP-10	2	43534535R42454323443	LUKOIL
800	15	0002MP-01	3	RT345344RE3222234675	MAGAZIN01
800	16	0003MP-02	4	55345345343453453455	10
800	17	2001MP-05	2	23434434123245342223	ADEF

How to add standard functions – CL_SALV_FUNCTIONS

Next, add functions to the application toolbar. For this, use the CL_SALV_FUNCTIONS class. Create the object reference variable and receive the object using the GET_FUNCTIONS method of the GR_TABLE object. Call the method SET_ALL to force the ALV grid to show all standard functions. Just add next code to your previous report:

```
DATA: gr_functions TYPE REF TO cl_salv_functions.

  gr_functions = gr_table->get_functions( ).
  gr_functions->set_all( abap_true ).
```

Result:

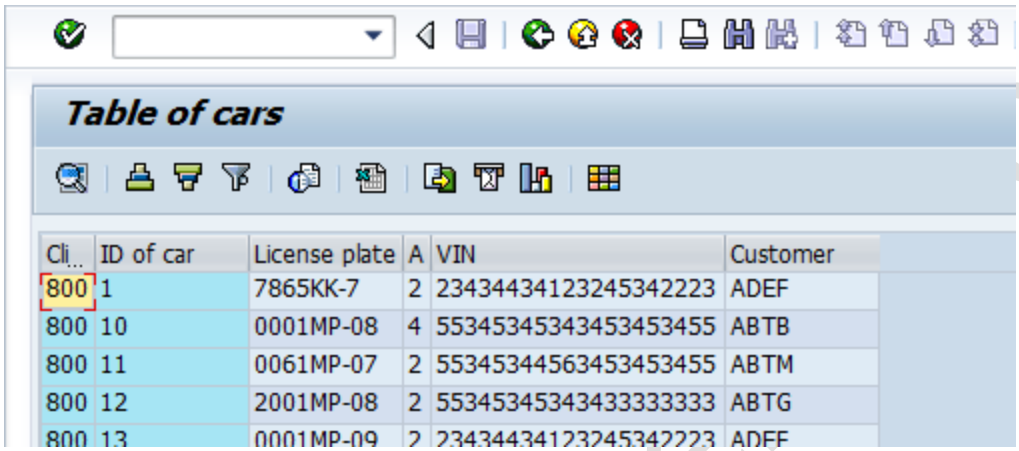


How to change display settings – CL_SALV_DISPLAY_SETTINGS

Next, we can change some display settings using the class CL_SALV_DISPLAY_SETTINGS. Create the object reference variable and receive the object using the GET_DISPLAY_SETTINGS method of the GR_TABLE object. In this example, we are setting the “Striped Pattern” for the ALV Grid rows, and setting the heading in the title bar.

```
DATA: gr_display TYPE REF TO cl_salv_display_settings.

gr_display = gr_table->get_display_settings( ).
gr_display->set_striped_pattern( cl_salv_display_settings=>true ).
gr_display->set_list_header( 'Table of cars' ).
```

Result:



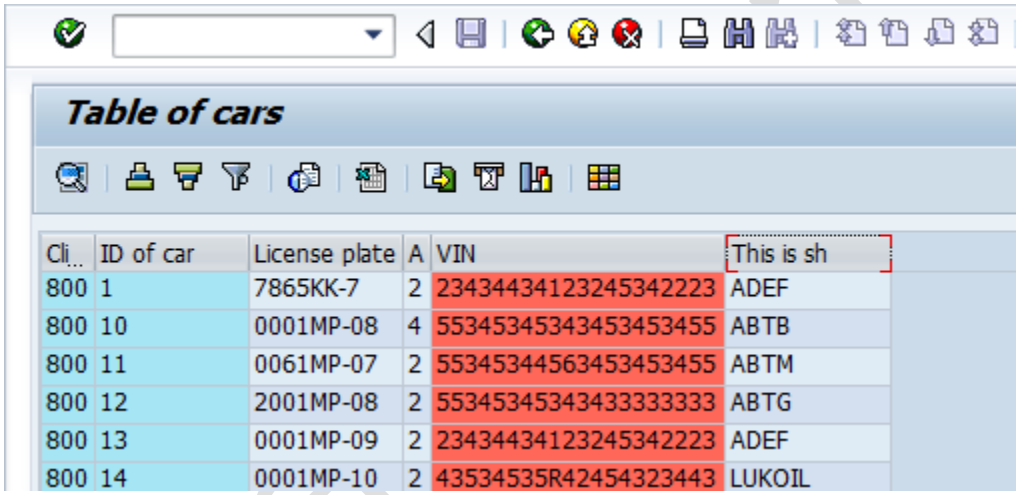
LeverX International, LLC
7A Lopatina str. 3 floor, Minsk, Belarus
Tel +375 17 232-20-01 http://www.leverx.by

How to manage columns – CL_SALV_COLUMNS_TABLE and CL_SALV_COLUMN_TABLE

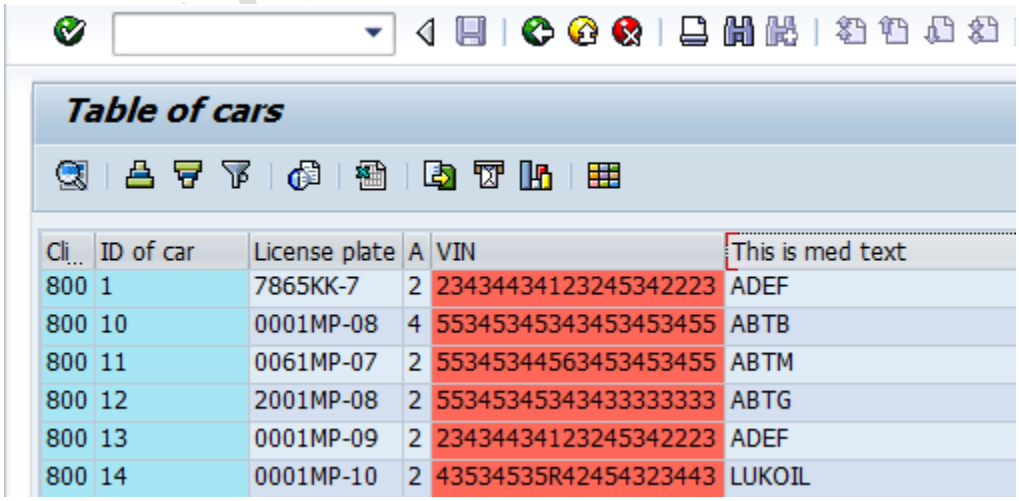
Next, we can change some of the attributes of a specific column in the ALV grid. In this example we will change the Heading Text of a column as well as the color of a column. Create the object reference variable and receive the object using the GET_COLUMNS method of the GR_TABLE object. This will pass you the object for all columns of the ALV grid. To access just one column, call the method GET_COLUMN from the GR_COLUMNS object. In this example, we are accessing the CUSTOMER column and the VIN column.

```
DATA: gr_columns TYPE REF TO cl_salv_columns_table.
DATA: gr_column TYPE REF TO cl_salv_column_table.
DATA: color TYPE lvc_s_colo.

 gr_columns = gr_table->get_columns( ).
 gr_column ?= gr_columns->get_column( 'CUSTOMER' ).
 gr_column->set_long_text( 'This is long text' ).
 gr_column->set_medium_text( 'This is med text' ).
 gr_column->set_short_text( 'This is sh' ).
 gr_column ?= gr_columns->get_column( 'VIN' ).
 color-col = '6'.
 color-int = '1'.
 color-inv = '0'.
 gr_column->set_color( color ).
```

Result:



After resizing 'Customer' column you can see the medium text:

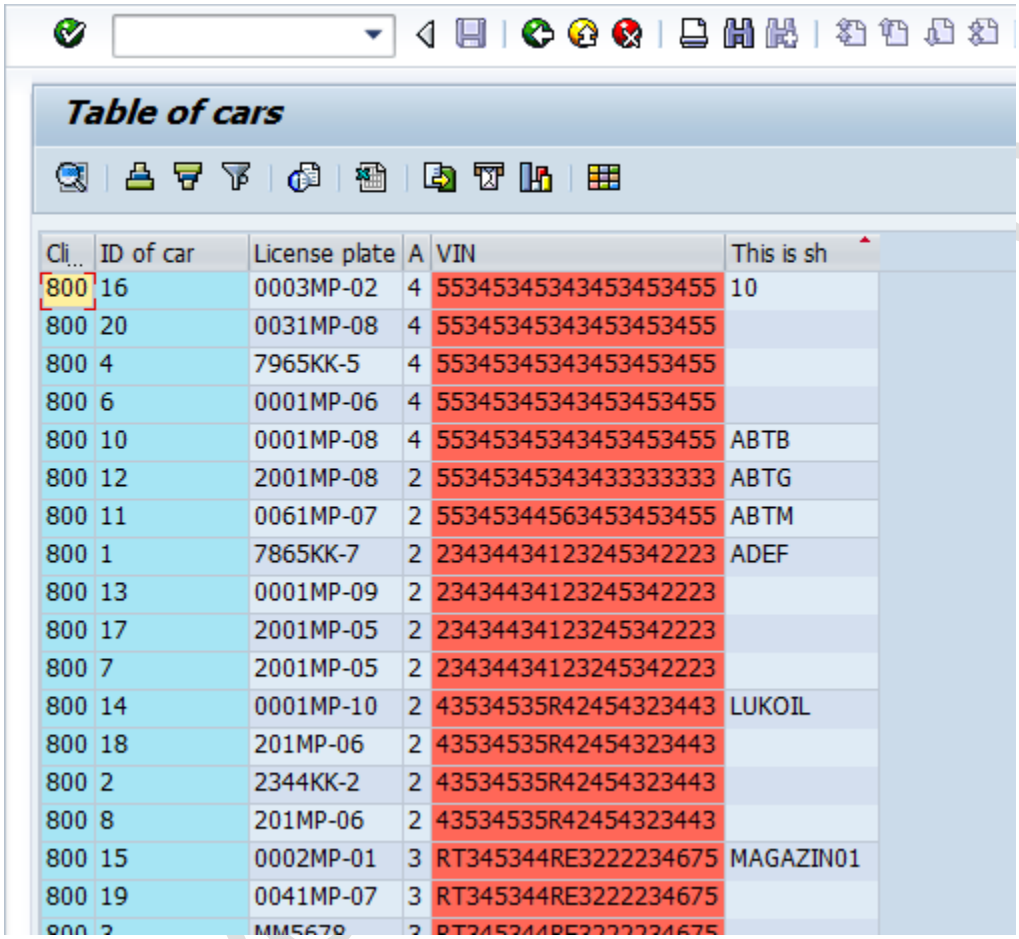


How to add sorts – CL_SALV_SORTS

Next, we can add some sorting to the ALV grid. Create the object reference variable and receive the object using the GET_SORTS method of the GR_TABLE object. Next, add the sort by calling the ADD_SORT method of the GR_SORTS object.

```
DATA: gr_sorts TYPE REF TO cl_salv_sorts.

  gr_sorts = gr_table->get_sorts( ).
  gr_sorts->add_sort( 'CUSTOMER' ).
```

Result:



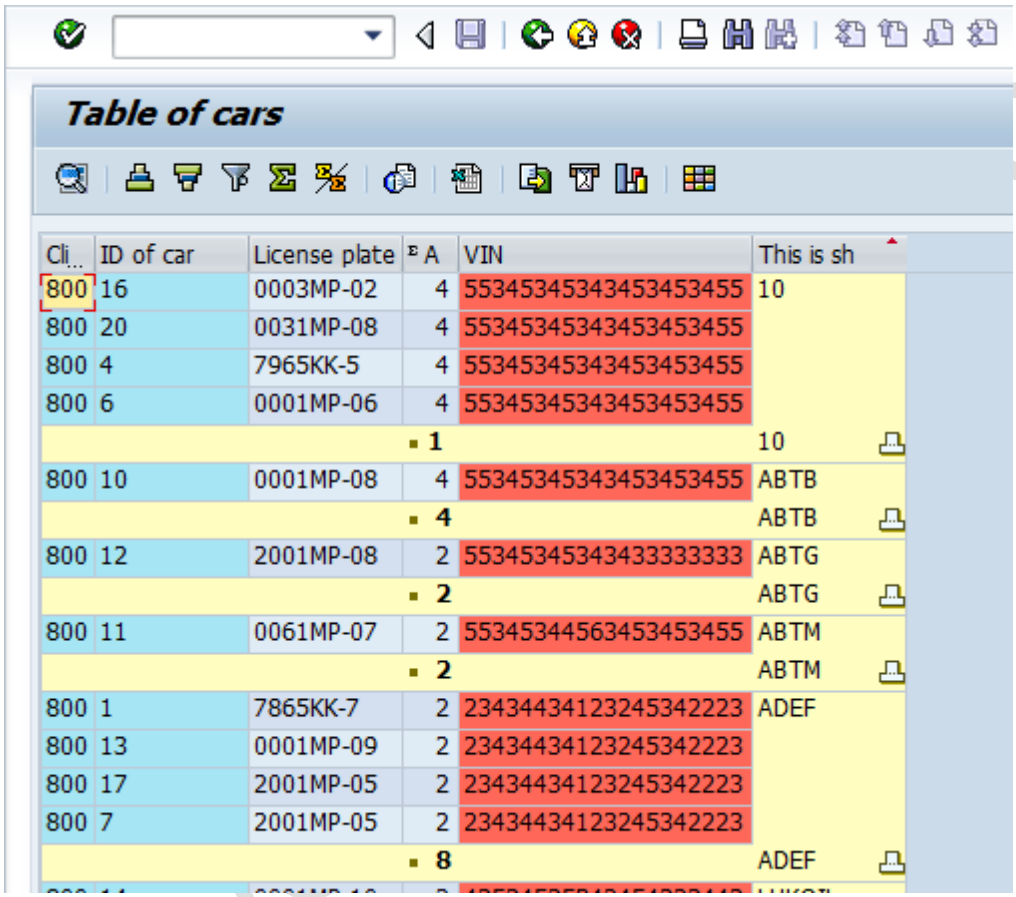
Cli...	ID of car	License plate	A	VIN	This is sh
800	16	0003MP-02	4	55345345343453453455	10
800	20	0031MP-08	4	55345345343453453455	
800	4	7965KK-5	4	55345345343453453455	
800	6	0001MP-06	4	55345345343453453455	
800	10	0001MP-08	4	55345345343453453455	ABTB
800	12	2001MP-08	2	55345345343433333333	ABTG
800	11	0061MP-07	2	55345344563453453455	ABTM
800	1	7865KK-7	2	23434434123245342223	ADEF
800	13	0001MP-09	2	23434434123245342223	
800	17	2001MP-05	2	23434434123245342223	
800	7	2001MP-05	2	23434434123245342223	
800	14	0001MP-10	2	43534535R42454323443	LUKOIL
800	18	201MP-06	2	43534535R42454323443	
800	2	2344KK-2	2	43534535R42454323443	
800	8	201MP-06	2	43534535R42454323443	
800	15	0002MP-01	3	RT345344RE3222234675	MAGAZIN01
800	19	0041MP-07	3	RT345344RE3222234675	

How to add aggregations – CL_SALV_AGGREGATIONS

Since we sorted by CUSTOMER, we can add an aggregation to subtotal the NAXES by CUSTOMER. Create the object reference variable and receive the object using the GET_AGGREGATIONS method of the GR_TABLE object. Next, add the aggregation by calling the ADD_AGGREGATION method of the GR_SORTS object. We also need to modify the call to ADD_SORT to set the SUBTOTAL = ABAP_TRUE.

```
DATA: gr_agg TYPE REF TO cl_salv_aggregations.

  gr_sorts->add_sort( columnname = 'CUSTOMER' subtotal = abap_true ).
  gr_agg = gr_table->get_aggregations( ).
  gr_agg->add_aggregation( 'NAXLES' ).
```

Result:



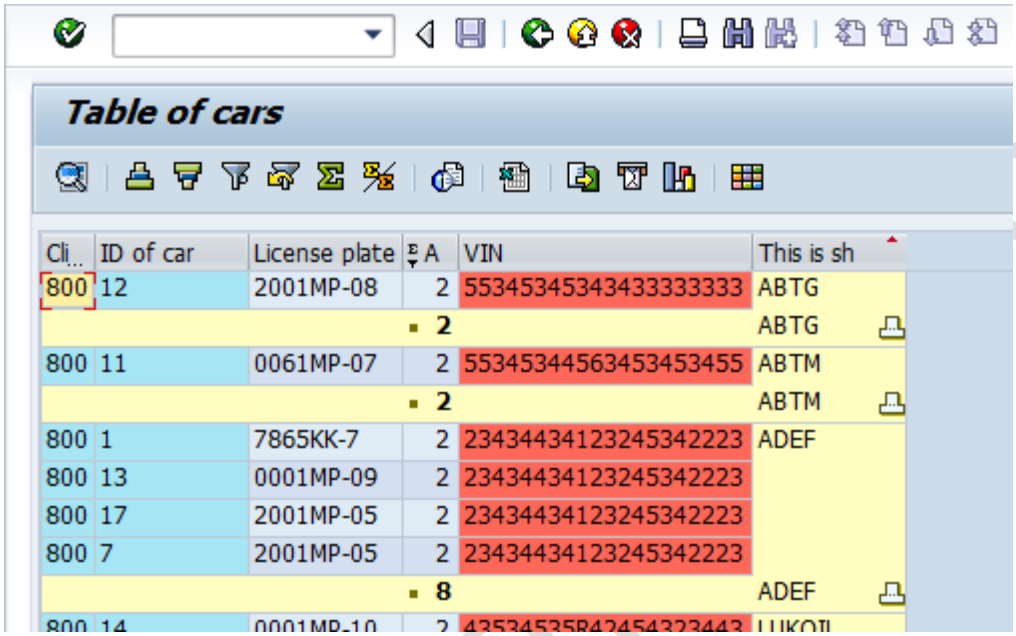
Cli...	ID of car	License plate	Σ A	VIN	This is sh
800	16	0003MP-02	4	55345345343453453455	10
800	20	0031MP-08	4	55345345343453453455	
800	4	7965KK-5	4	55345345343453453455	
800	6	0001MP-06	4	55345345343453453455	
			1		10
800	10	0001MP-08	4	55345345343453453455	ABTB
			4		ABTB
800	12	2001MP-08	2	55345345343433333333	ABTG
			2		ABTG
800	11	0061MP-07	2	55345344563453453455	ABTM
			2		ABTM
800	1	7865KK-7	2	23434434123245342223	ADEF
800	13	0001MP-09	2	23434434123245342223	
800	17	2001MP-05	2	23434434123245342223	
800	7	2001MP-05	2	23434434123245342223	
			8		ADEF

How to add filters – CL_SALV_FILTERS

Using the CL_SALV_FILTERS class we can setup some filters for the data in our ALV grid. Create the object reference variable and receive the object using the GET_FILTERS method of the GR_TABLE object, and then simply call the method ADD_FILTER with the parameters.

```
DATA: gr_filter TYE REF TO cl_salv_filters.

  gr_filter = gr_table->get_filters( ).
  gr_filter->add_filter( columnname = 'NAXLES' low = '2' ).
```

Result:



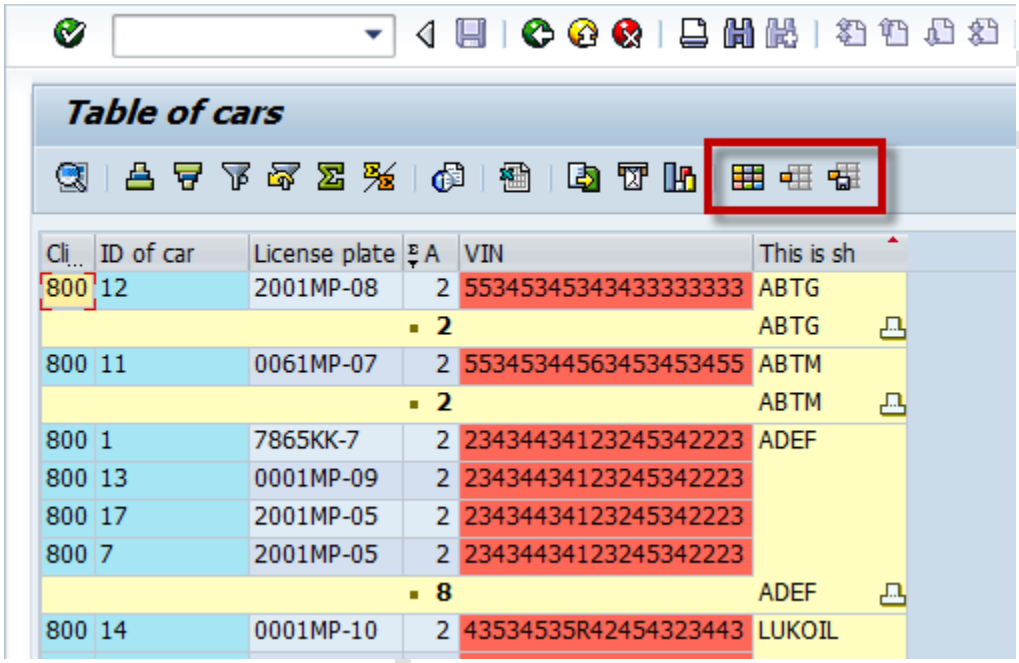
LeverX International, LLC
7A Lopatina str. 3 floor, Minsk, Belarus
Tel +375 17 232-20-01 http://www.leverx.by

How to manage layouts – CL_SALV_LAYOUT

If you want to allow the user to manage layouts of the ALV grid, you must use the class CL_SALV_LAYOUT. Create the object reference variable and receive the object using the GET_LAYOUT method of the GR_TABLE object. Then simply call the method SET_KEY with the parameters and set the save restriction using the SET_SAVE_RESTRICTION method.

```
DATA: gr_layout TYPE REF TO cl_salv_layout.
DATA: key TYPE salv_s_layout_key.

gr_layout = gr_table->get_layout( ).
key-report = sy-repid. gr_layout->set_key( key ).
gr_layout->set_save_restriction( cl_salv_layout=>restrict_none ).
```

Result:



LeverX International, LLC
7A Lopatina str. 3 floor, Minsk, Belarus
Tel +375 17 232-20-01 http://www.leverx.by

Picture control and Drag & Drop functionality

In this unit we will create simple report to demonstrate main principles of picture and drag & drop controls.

Picture control

Use

Picture control allows a user to show a picture at screen.

Drag & Drop

Use

Drag and drop allows the user to select an object from one part of a custom control (source) and drop it on another part of a custom control (target). An action occurs in the second part that depends on the object type. Source and target may be either the same control or different controls.

Prerequisites

For a control to support drag and drop, the control wrapper must provide drag and drop events. You must then write handler methods for these events in your program. The events are registered automatically by the relevant control wrapper.

Features

A particular drag and drop behavior is set for each custom control. This behavior may be set globally for all elements of the control (for example, SAP Textedit), or you may be able to define a different behavior for each component (for example SAP Tree). Each behavior consists of one or more descriptions.

A description has the following attributes:

- **DragSrc:** Object is the source of a drag and drop procedure
- **DropTarget:** Object is the target of a drag and drop procedure
- **Flavor:** The flavor describes the type of a drag and drop description. In a drag and drop operation, you can only drop an object onto another if both have at least one common description.
- **Effect:** Specifies whether the drag and drop operations copies or moves the object.
- **Effect_In_Ctrl:** The drop effect used when you copy or move data within the same control.

As soon as a drag event is triggered, you must use the corresponding handler method to find out the affected object.

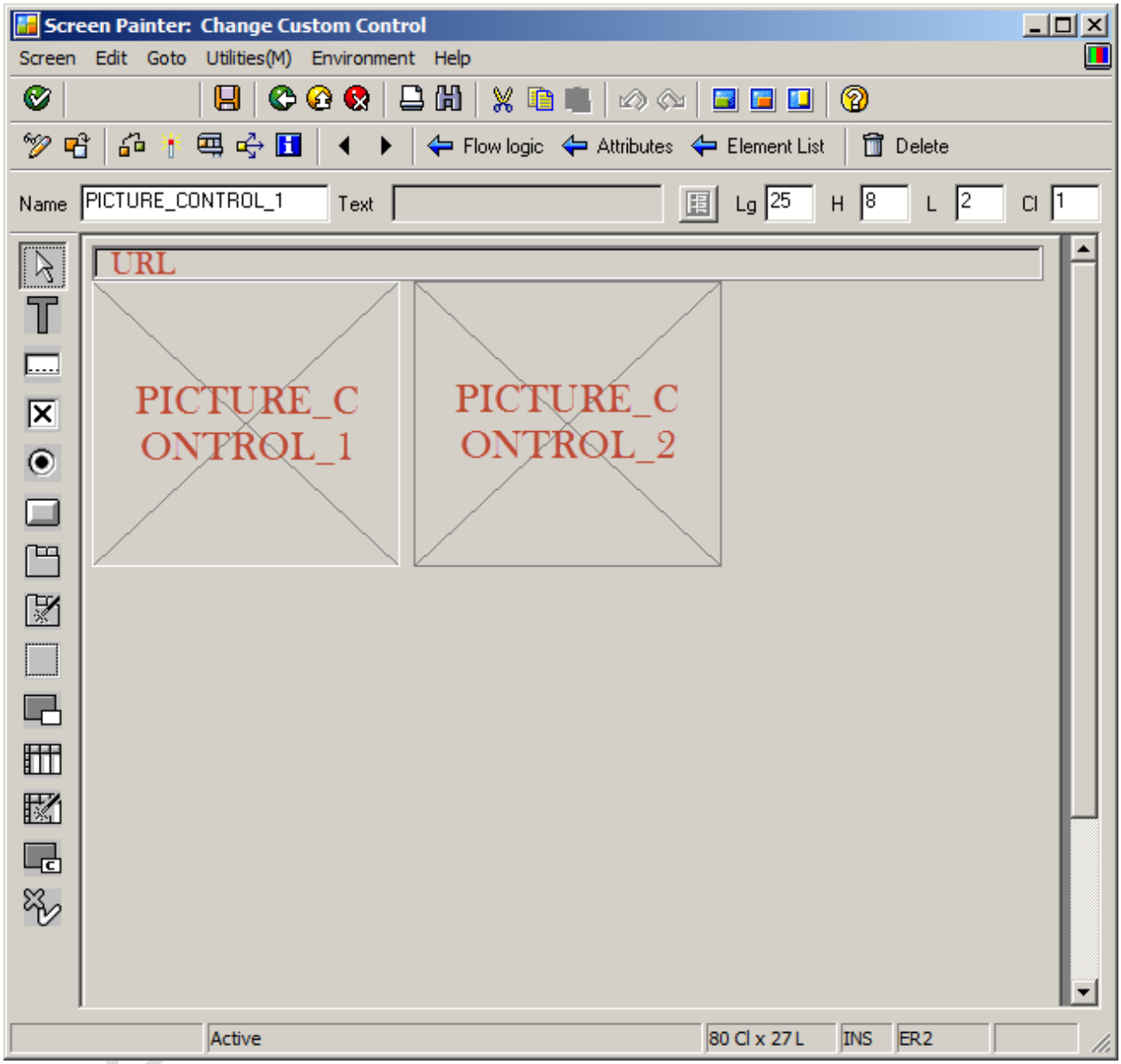
You must also define the action that is to be carried out on the drop event. The action usually depends on the object that you drop in the control.

If you assign more than one flavor to an object, you must define which flavor is to be used. You do this in the handler for another event.

Once the drop event is finished, you can use a further event to implement additional actions. This is particularly useful for deleting the dropped object from the source after a move operation.

How to create a picture control on a custom screen

First we need to do for our picture controls is to create custom screen (we used SCREEN NUMBER 200 in this example, but you can use different) with two Custom Container controls and one Input Field control. We will load our pictures into custom containers:



URL field is used to show image link, which will be generated later. This field is optional.

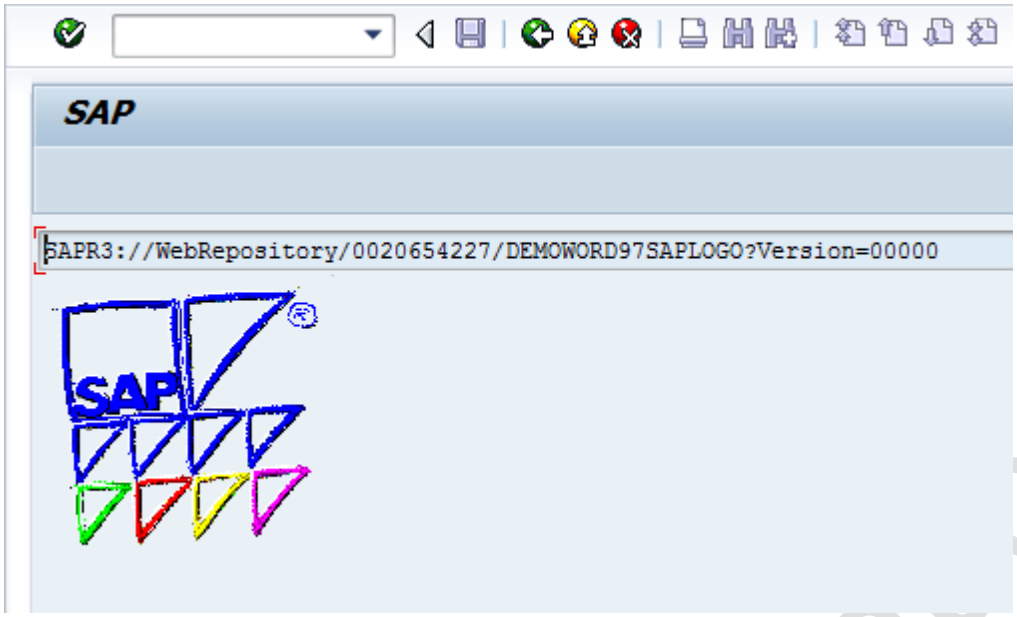
LeverX International, LLC
7A Lopatina str. 3 floor, Minsk, Belarus
Tel +375 17 232-20-01 http://www.leverx.by

After creation of custom screen we must define screen elements and load a picture into first container:

```
REPORT  zlx_cw7_2.

SET SCREEN 200.

TYPE-POOLS cndp. "Declaration for use of custom conainers

DATA url TYPE cndp_url.
DATA picture_control_1 TYPE REF TO cl_gui_picture.
DATA picture_control_2 TYPE REF TO cl_gui_picture.
DATA container_1 TYPE REF TO cl_gui_custom_container.
DATA container_2 TYPE REF TO cl_gui_custom_container.
DATA event_tab TYPE cntl_simple_events.

*Define custom containers
CREATE OBJECT container_1
  EXPORTING
    container_name = 'PICTURE_CONTROL_1'.

CREATE OBJECT container_2
  EXPORTING
    container_name = 'PICTURE_CONTROL_1'.

*Define picturea object and set it to container
CREATE OBJECT picture_control_1
  EXPORTING
    parent = container_1.

CREATE OBJECT picture_control_2
  EXPORTING
    parent = container_2.

* Register the events for use in drag'n'drop
CALL METHOD picture_control_1->set_registered_events
  EXPORTING
    events = event_tab.

CALL METHOD picture_control_2->set_registered_events
  EXPORTING
    events = event_tab.

*Generate link for picture
CALL FUNCTION 'DP_PUBLISH_WWW_URL'
  EXPORTING
    objid    = 'DEMOWORD97SAPLOGO'
    lifetime = cndp_lifetime_transaction
  IMPORTING
    url      = url.

*Show picture
CALL METHOD picture_control_1->load_picture_from_url_async
  EXPORTING
    url = url.
```

LeverX International, LLC
7A Lopatina str. 3 floor, Minsk, Belarus
Tel +375 17 232-20-01 http://www.leverx.by

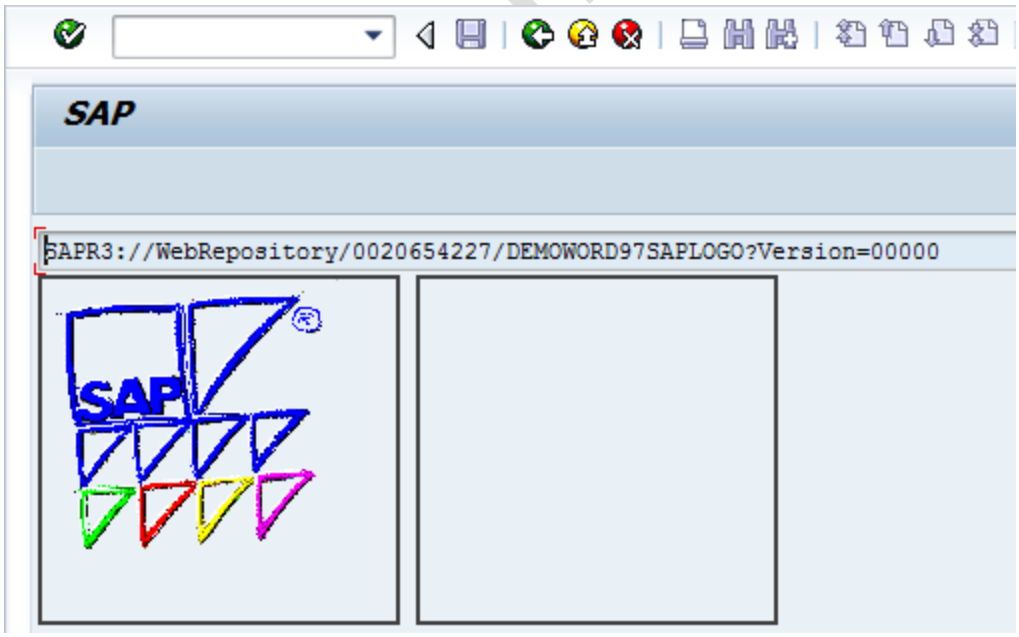
When you will execute your report you will see something like this:



Also you can add 3D border for the containers for easier drag & drop actions:

```
* Set 3D Border for containers
CALL METHOD picture_control_1->set_3d_border
  EXPORTING
    border = 1.
CALL METHOD picture_control_2->set_3d_border
  EXPORTING
    border = 1.
```

Result:



LeverX International, LLC
7A Lopatina str. 3 floor, Minsk, Belarus
Tel +375 17 232-20-01 http://www.leverx.by

How to create a Drag & Drop functionality

Next we need to create Drag & Drop functionality to drag picture from contaier 1 to container 2.

First you need to define 2 local classes (you must do definition of local classes before the DATA section):

```
class c_event_receiver definition.
* The class is used to test the events raised by the cl_gui_picture
  public section.
    methods event_handler_ondrag
          for event ondrag of cl_gui_picture importing DRAGDROPOBJ.
    methods event_handler_ondrop
          for event ondrop of cl_gui_picture
          importing DRAGDROPOBJ sender.
endclass.

* The class is used to store image link
class lcl_dragdrop_data definition.
  public section.
    data: m_url(255) type c.
endclass.
```

C_EVENT_RECEIVER class will be used to handle events ONDARAG and ONDROP from CL_GUI_PICTURE class. LCL_DRAGDOP_DATA class contains only one variable m_url with type of char. This variable will store link for image, which will be copied from PICTURE_CONTROL_1 on ONDRAG event and after that, this value will be sent to PICTURE_CONTROL_2 on ONDROP event. We need to create local class lcl_dragdrop_data, because impoting parameter ONDRAGOBJ of CL_GUI_PICTURE class must be type of OBJECT.

LeverX International, LLC
7A Lopatina str. 3 floor, Minsk, Belarus
Tel +375 17 232-20-01 http://www.leverx.by

Next we need to define data and objects:

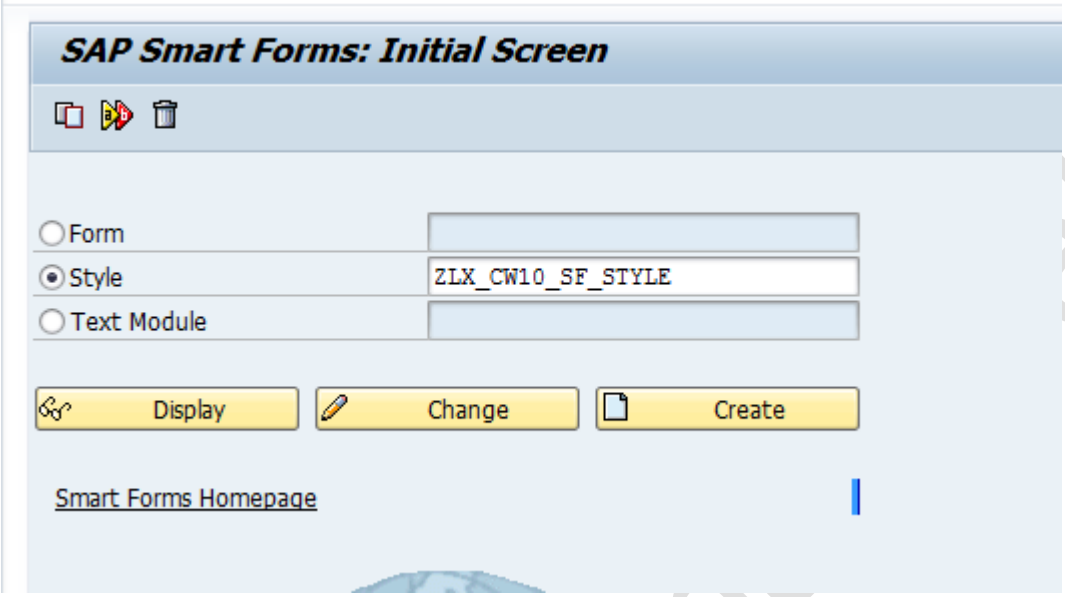
```
DATA event_receiver  TYPE REF TO c_event_receiver.
DATA: dragdrop_1 TYPE REF TO cl_dragdrop.
DATA: dragdrop_2 TYPE REF TO cl_dragdrop.

* Create the event_receiver object and set the handlers for the events
* of the picture controls
CREATE OBJECT event_receiver.
SET HANDLER event_receiver->event_handler_ondrag
        FOR picture_control_1.
SET HANDLER event_receiver->event_handler_ondrop
        FOR picture_control_2.

* Drag&Drop
CREATE OBJECT dragdrop_1.
CREATE OBJECT dragdrop_2.
CALL METHOD dragdrop_1->add
  EXPORTING
    flavor     = 'Image'                          "#EC NOTEXT
    dragsrc    = 'X'
    droptarget = space
    effect     = cl_dragdrop=>copy.

CALL METHOD dragdrop_2->add
  EXPORTING
    flavor     = 'Image'                          "#EC NOTEXT
    dragsrc    = space
    droptarget = 'X'
    effect     = cl_dragdrop=>copy.

*Set dragdrop1 objec to picture 1. It means that picture 1 will be a source and we can only drag it
CALL METHOD picture_control_1->set_dragdrop_picture
  EXPORTING
    dragdrop = dragdrop_1.

*Set dragdrop1 objec to picture 1. It means that picture 2 will be a target
and we can only drop source on it
CALL METHOD picture_control_2->set_dragdrop_control
  EXPORTING
    dragdrop = dragdrop_2.
CALL METHOD picture_control_2->set_dragdrop_picture
  EXPORTING
    dragdrop = dragdrop_2. "set picture same as dragdrop source.
```

And final we also need to implement our local classes (you must implement your local classes at the end of your source code):

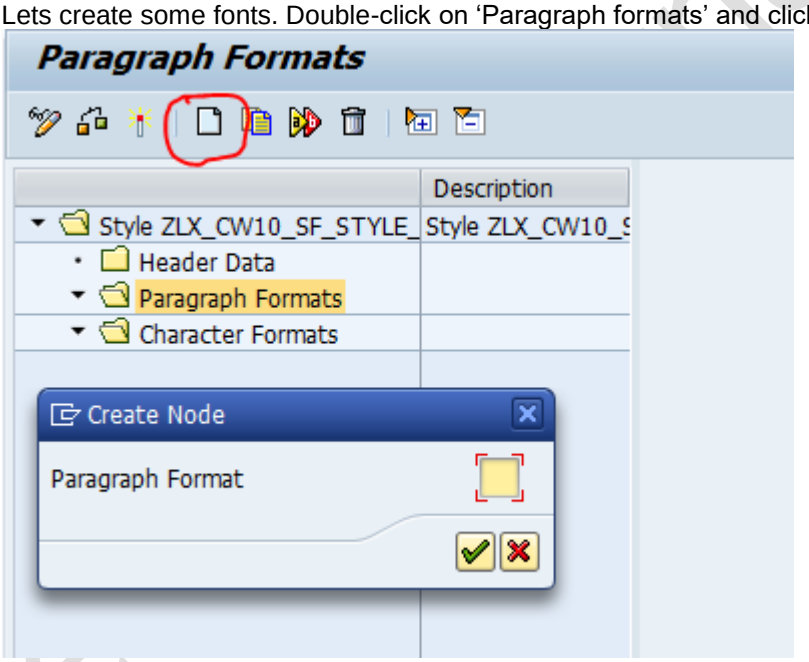
```
CLASS c_event_receiver IMPLEMENTATION.

  METHOD event_handler_ondrag.

    DATA dd_data TYPE REF TO lcl_dragdrop_data.
    CREATE OBJECT dd_data.
    dd_data->m_url = url. " when PICURE_CONROL_1 raises event ONDRAG
                          " image link copies into object dd_data
    dragdropobj->object = dd_data.

  ENDMETHOD.                "EVENT_HANDLER_ondrag

  METHOD event_handler_ondrop.
    DATA dd_data TYPE REF TO lcl_dragdrop_data.
    dd_data ?= dragdropobj->object.

    CALL METHOD sender->load_picture_from_url
      EXPORTING
        url = dd_data->m_url. " when PICTURE_CONROL_2 raises event
                              " ONDROP image link will be copied
                              " from dd_data object

  ENDMETHOD.                "EVENT_HANDLER_ondrop

ENDCLASS.                 "C_event_receiver IMPLEMENTATION
```

How to create a Smart Form

1. Go to transaction SMARTFORMS.
2. In this transaction you can create 'Style'. This tool intended for tuning fonts and indents.
3. Create new Style:

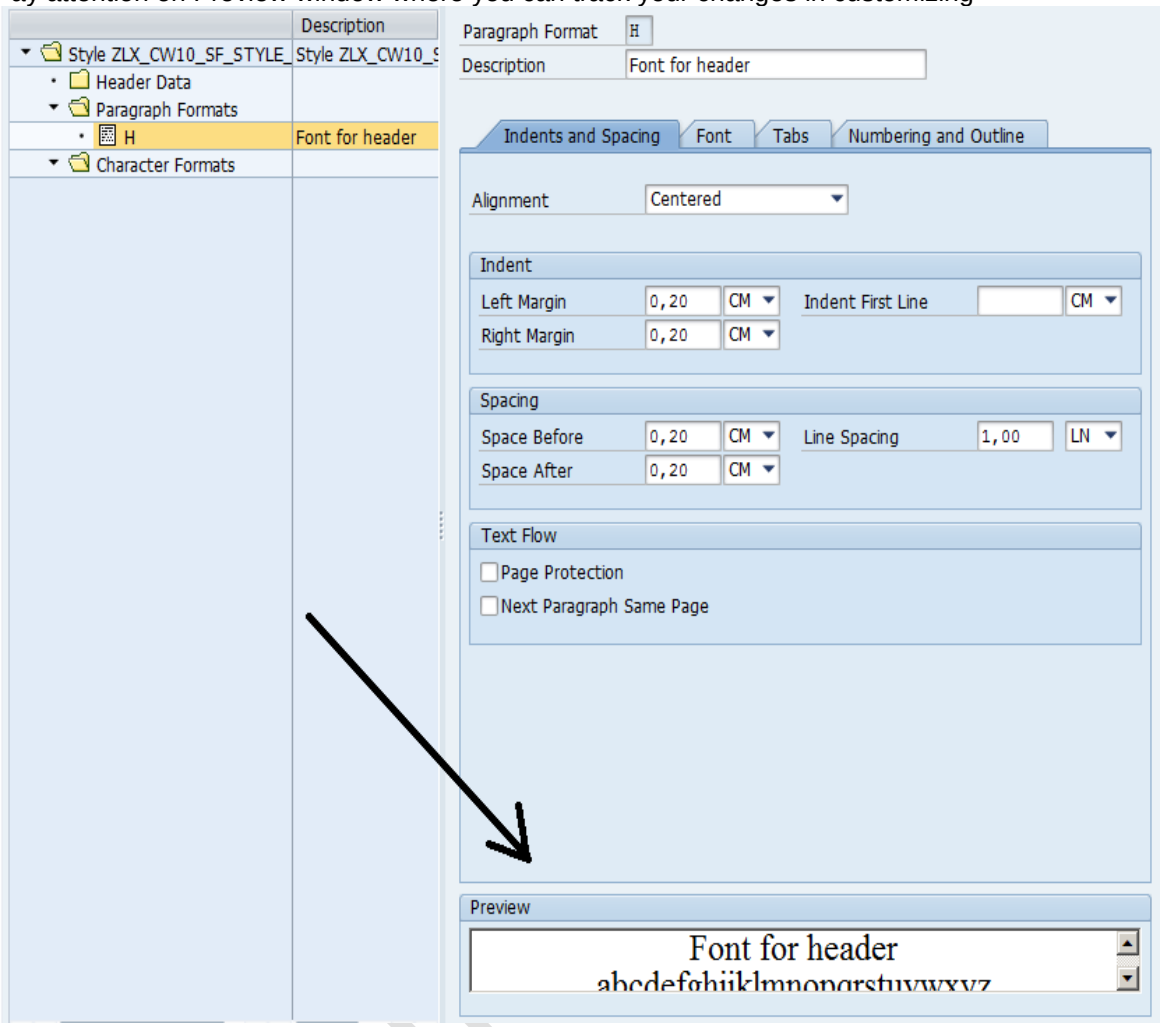


4. Lets create some fonts. Double-click on 'Paragraph formats' and click 'Create node' (F5):

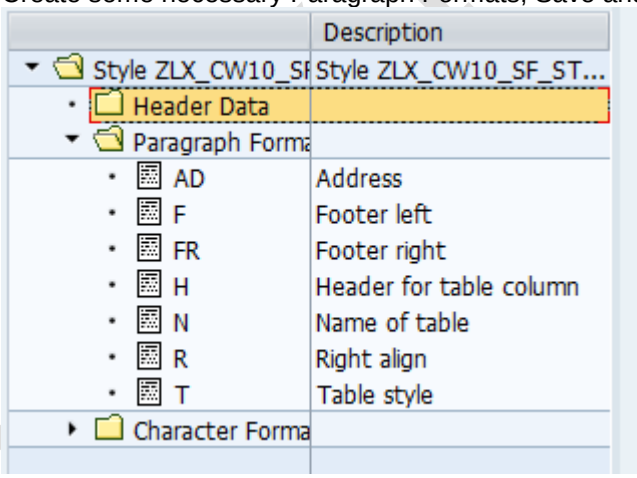


5. Write explanatory symbol (e.g. 'H' for header font or 'T' for text in table cells), press Enter.
6. In tabs 'Indents and Spacing' and 'Font' you can customize fonts settings and indents.

7. Pay attention on Preview window where you can track your changes in customizing

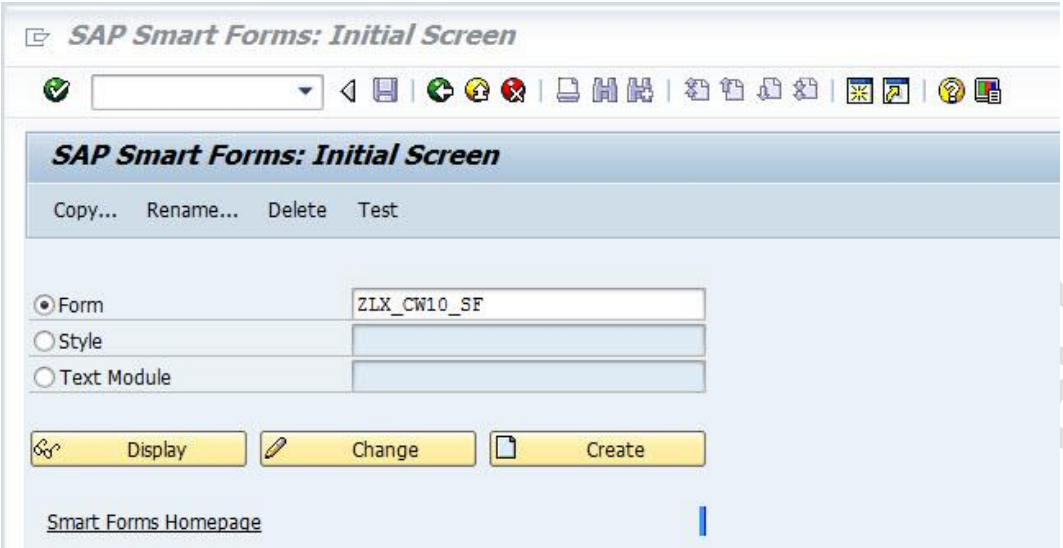


8. Create some necessary Paragraph Formats, Save and Activate it.

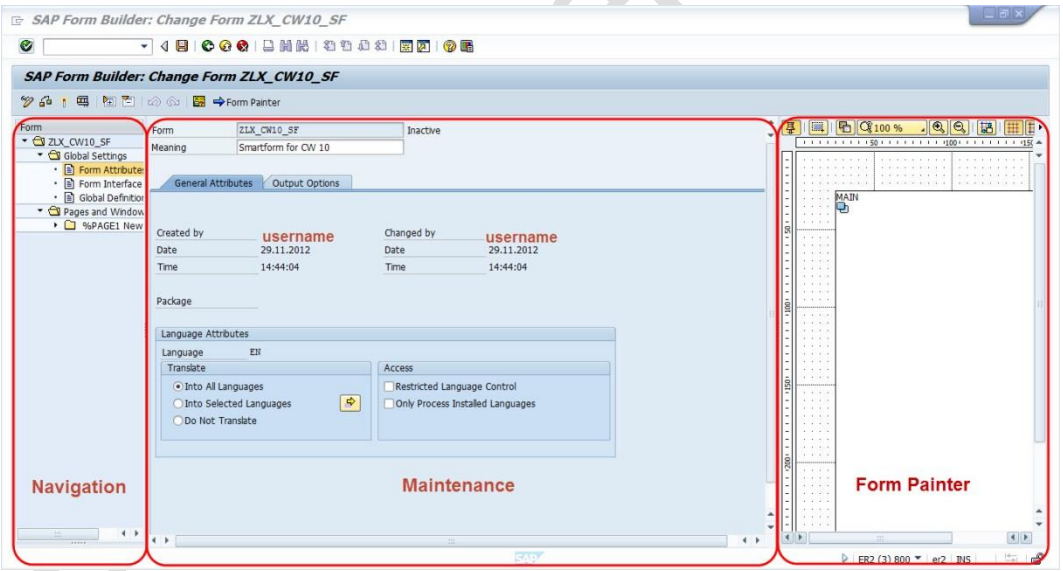


9. Go to transaction SMARTFORMS.

10. Enter a name for the form and click on Create.



11. Enter a short description for the form.
    The screen is divided into three sections:
    a. **Navigation window** consist of nodes and sub nodes. They contain all the elements (text, window etc.) that belong to sap forms
    b. **Maintenance window** shows attributes of the elements
    c. **Form printer window** shows the layout of the page

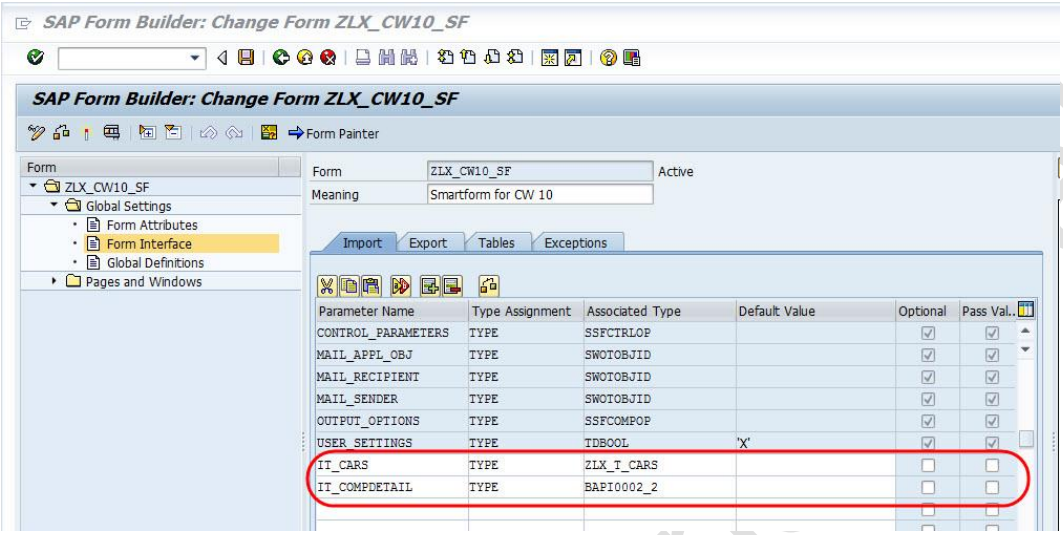


In the navigation window you can see automatically created a tree structure which contains:
    a. **Global settings**
        i. Form Attributes
        ii. Form Interface
        iii. Global Definitions
    b. **Pages and windows**
        i. First page
        ii. Main window

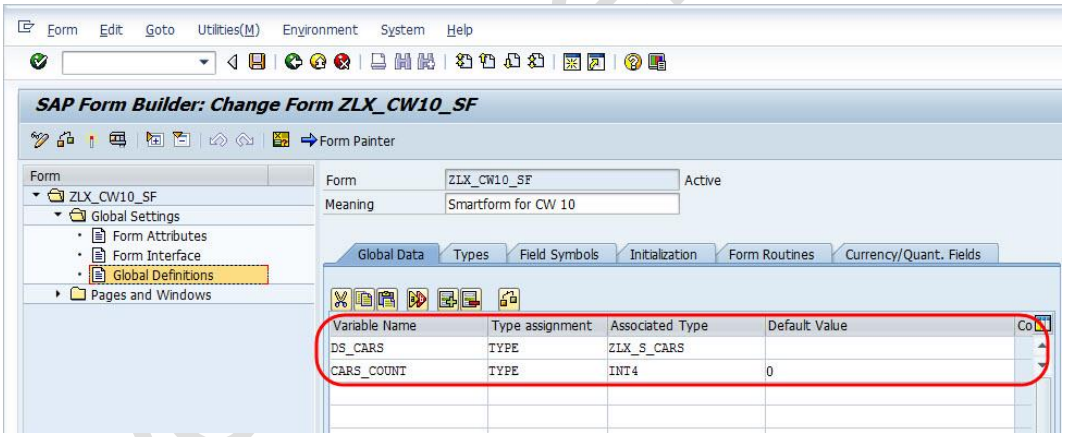
12. Add variables to Global Settings.
    In FORM ATTRIBUTES you have two tab pages: General Attributes and Output options.
    In FORM INTERFACE you have four tab pages:

a. Import — parameters that has to be imported to the form function module.
b. Export — parameters that are to be exported from the form function module.
c. Tables — parameters that are used to pass internal tables form the driver program.
d. Exceptions — exceptions that are created in the function module.

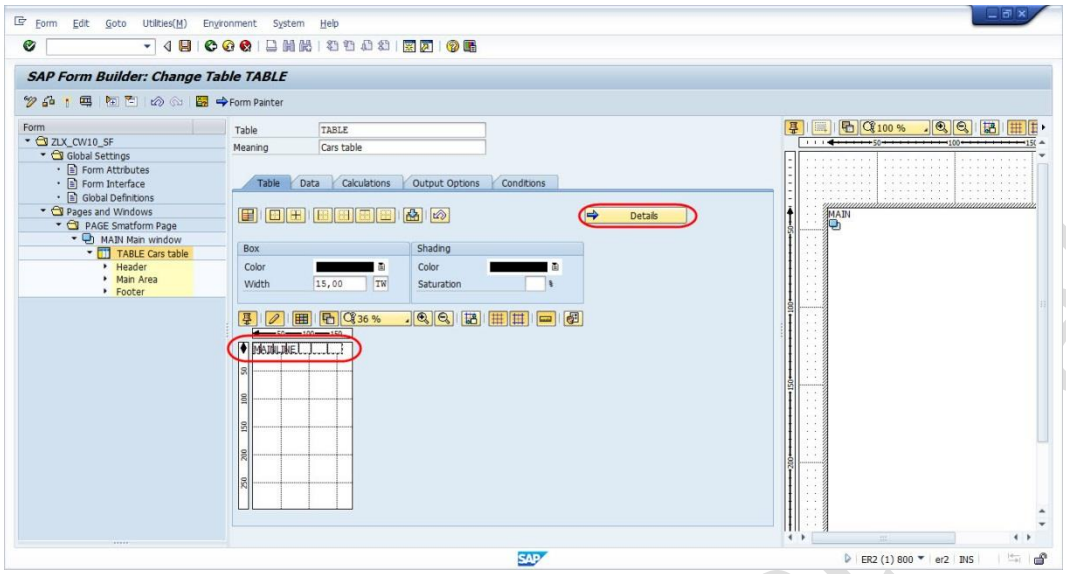
<u>All the parameters that are defined in the form interface will be displayed as parameters in the form function module.</u>



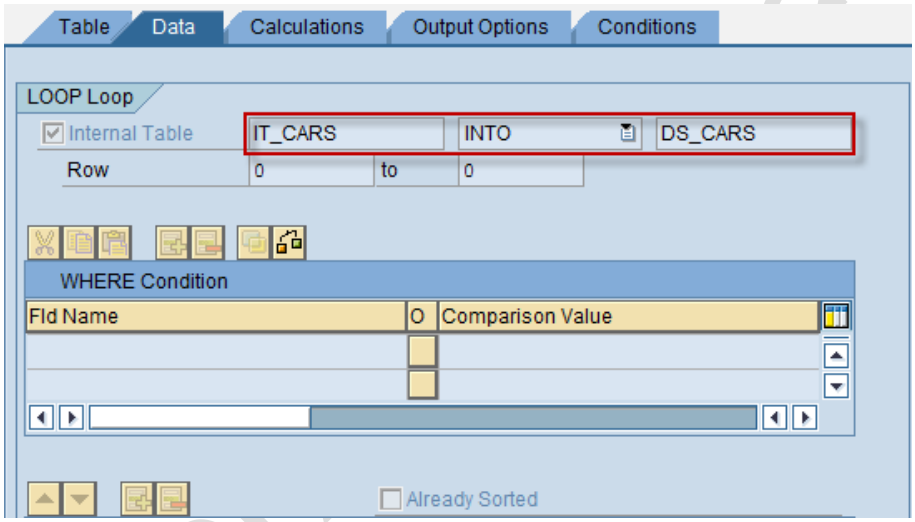
In GLOBAL DEFINITIONS you can define global variables, types, field symbols, etc. That is global to the smartform.



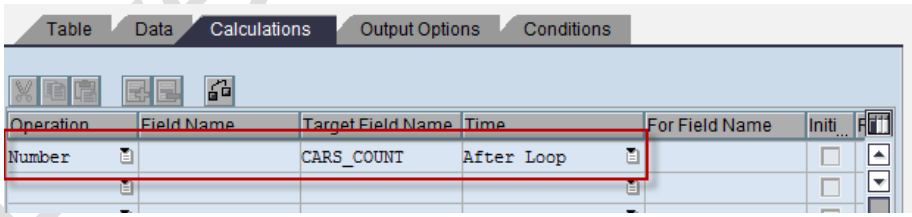
13. Double click on *Pages and Windows*. Right click on *Main Window*, create table. Provide Table name and Meaning. On the tab *Table* add cells to line. You can rename line in *Details.*



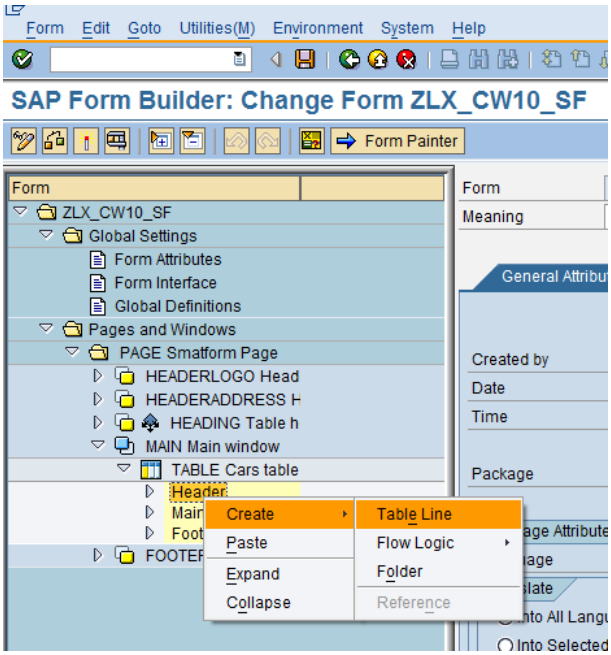
Select tab *Data* and add next information:



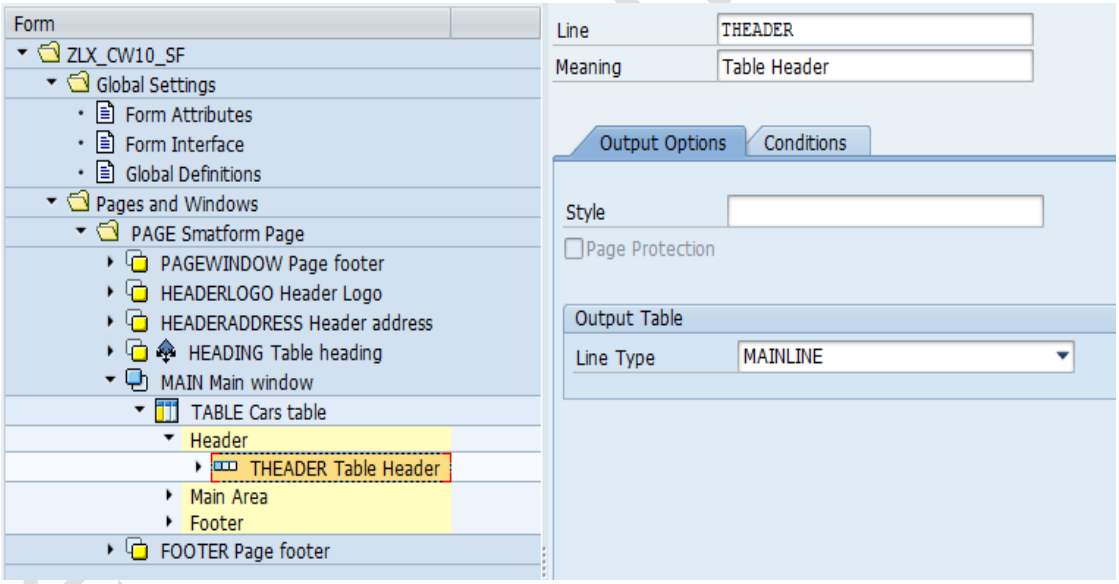
Select tab *Calculations* and add next information:



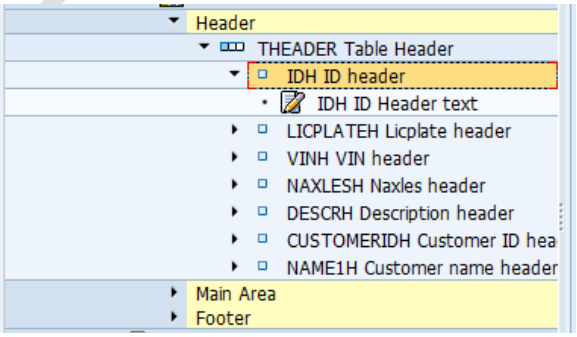
14. Right click on *Header*, *Create->Table Line.*



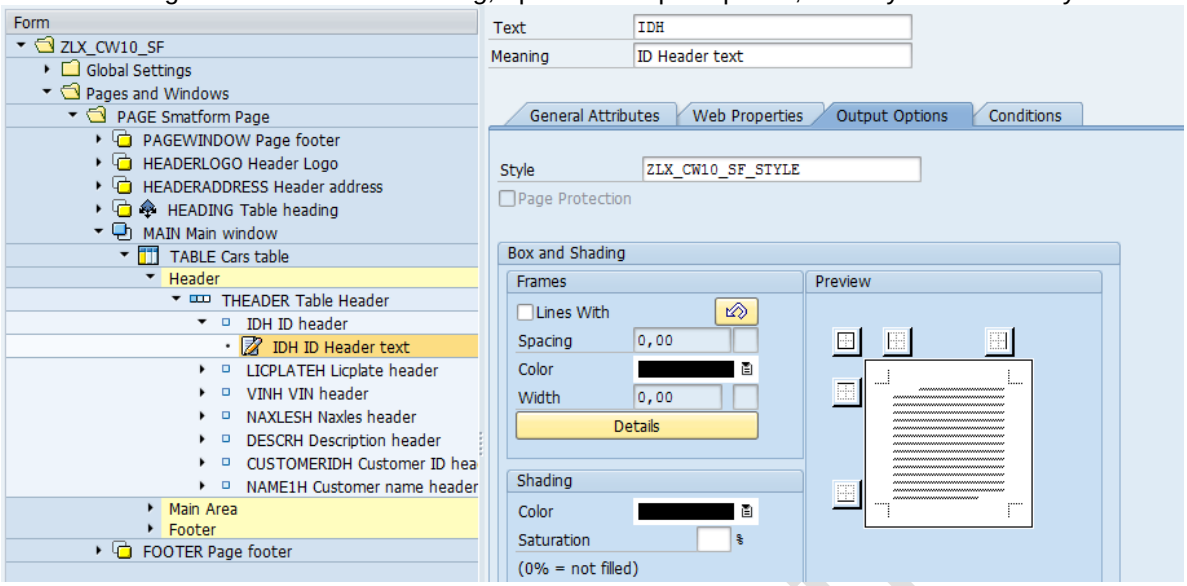
Choose Line Type, You can change *Line name* and *Meaning*



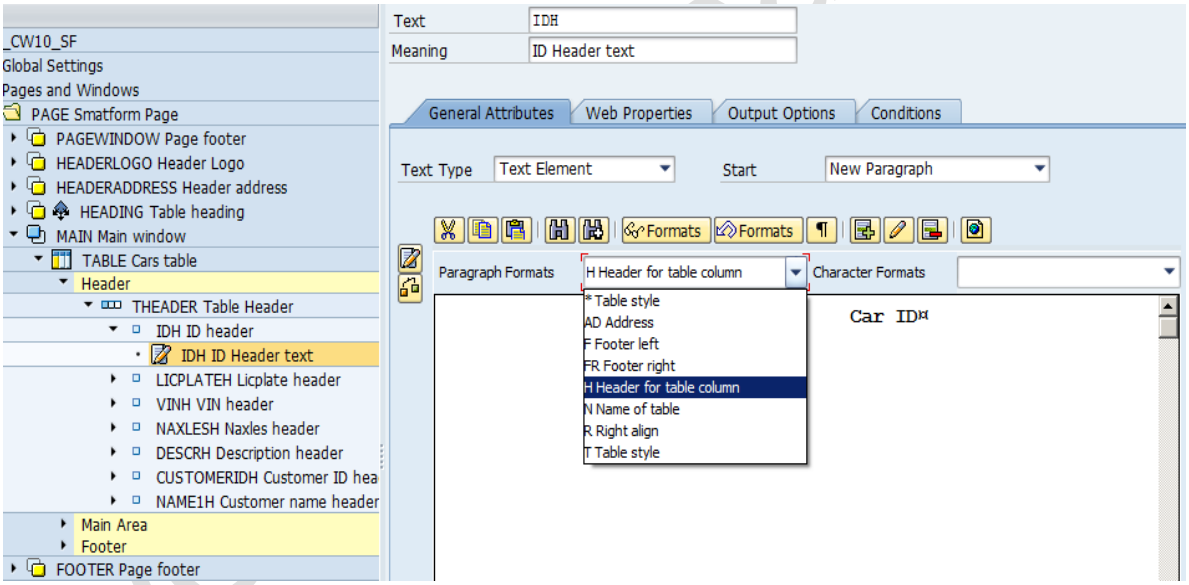
Rename cells name and meaning.



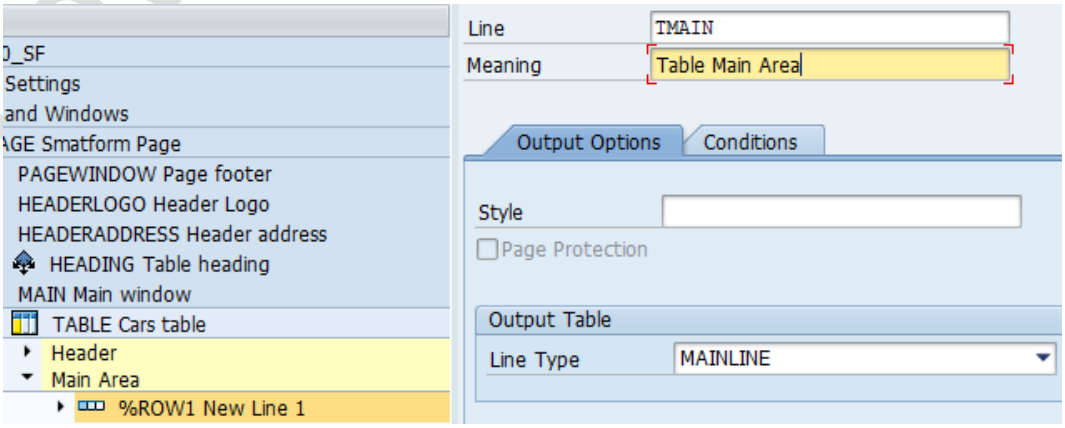
Right click on *cell-> Create->Text*
You can change *Text name* and *Meaning,* open tab Output Options, select your created style.



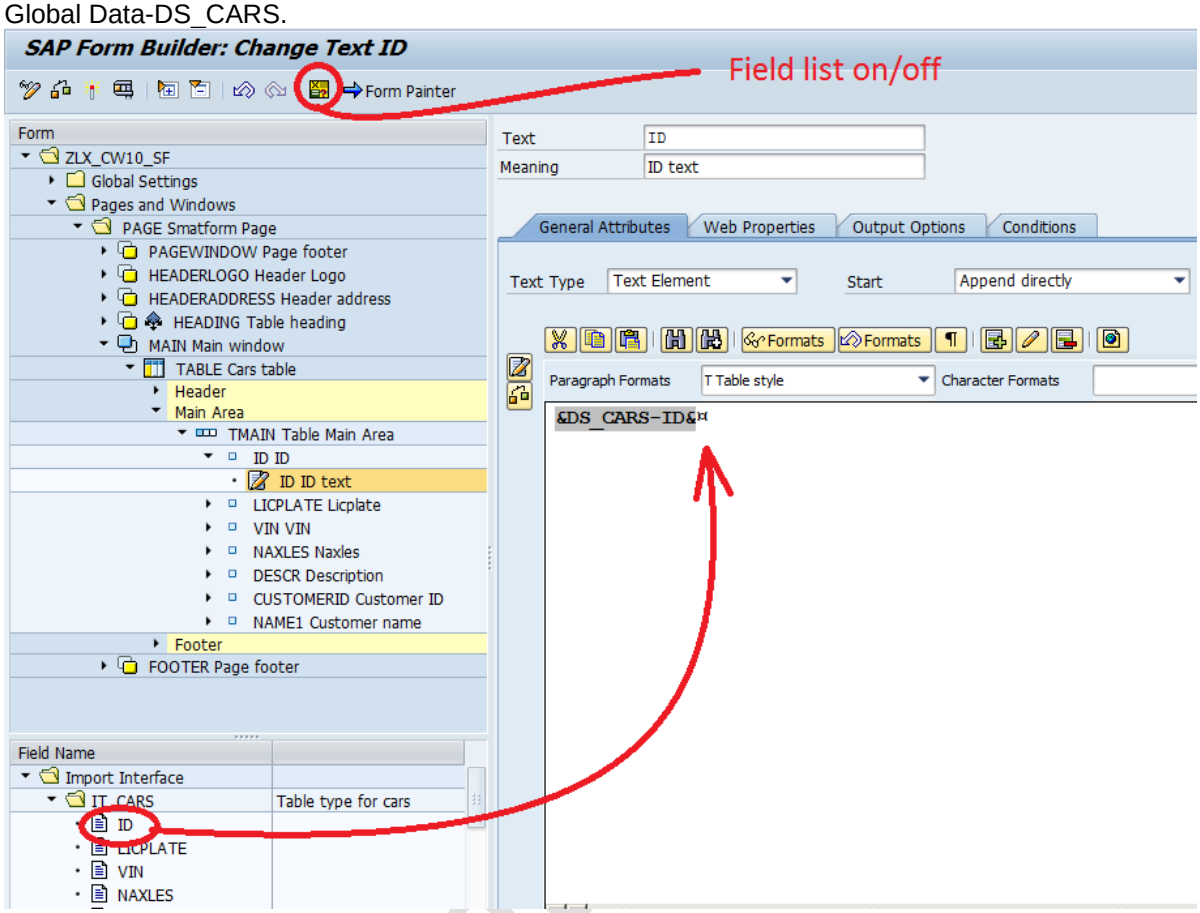
On the tab *General Attributes* choose Paragraph Format and enter a column name



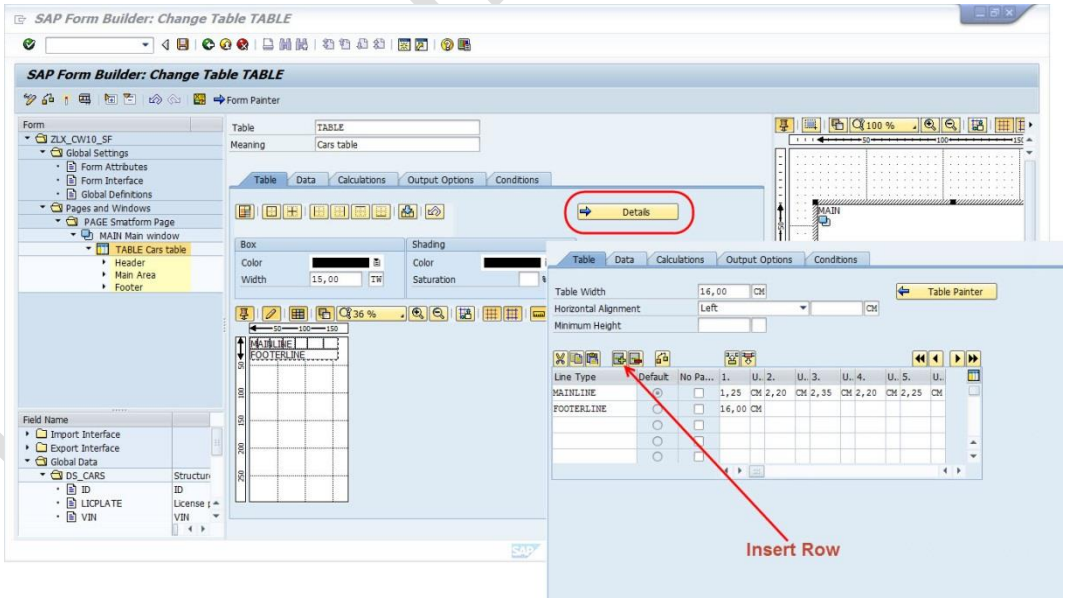
15. Create Line for Main Area.



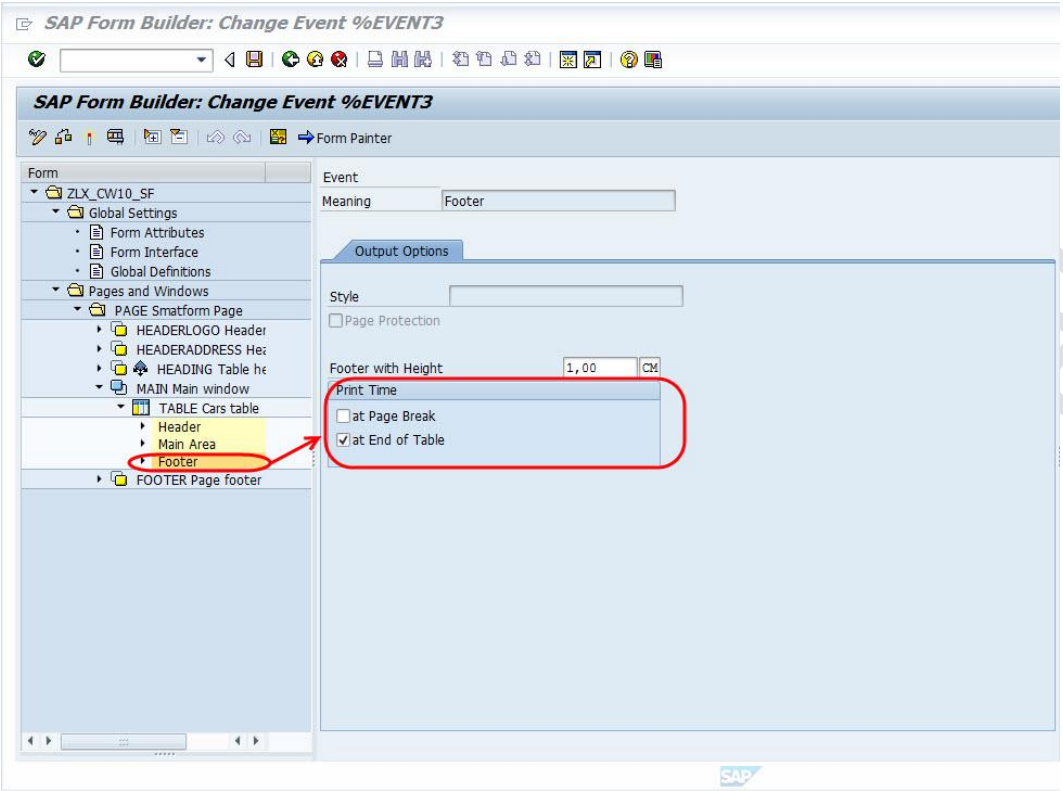
16. Create Text for Cells. Select Style in the tab ‘Output Options’, choose Paragraph format in ‘General Attributes’ tab. Select “Field List on/off”. Drag and Drop information to the text field from Field Name-Global Data-DS_CARS.



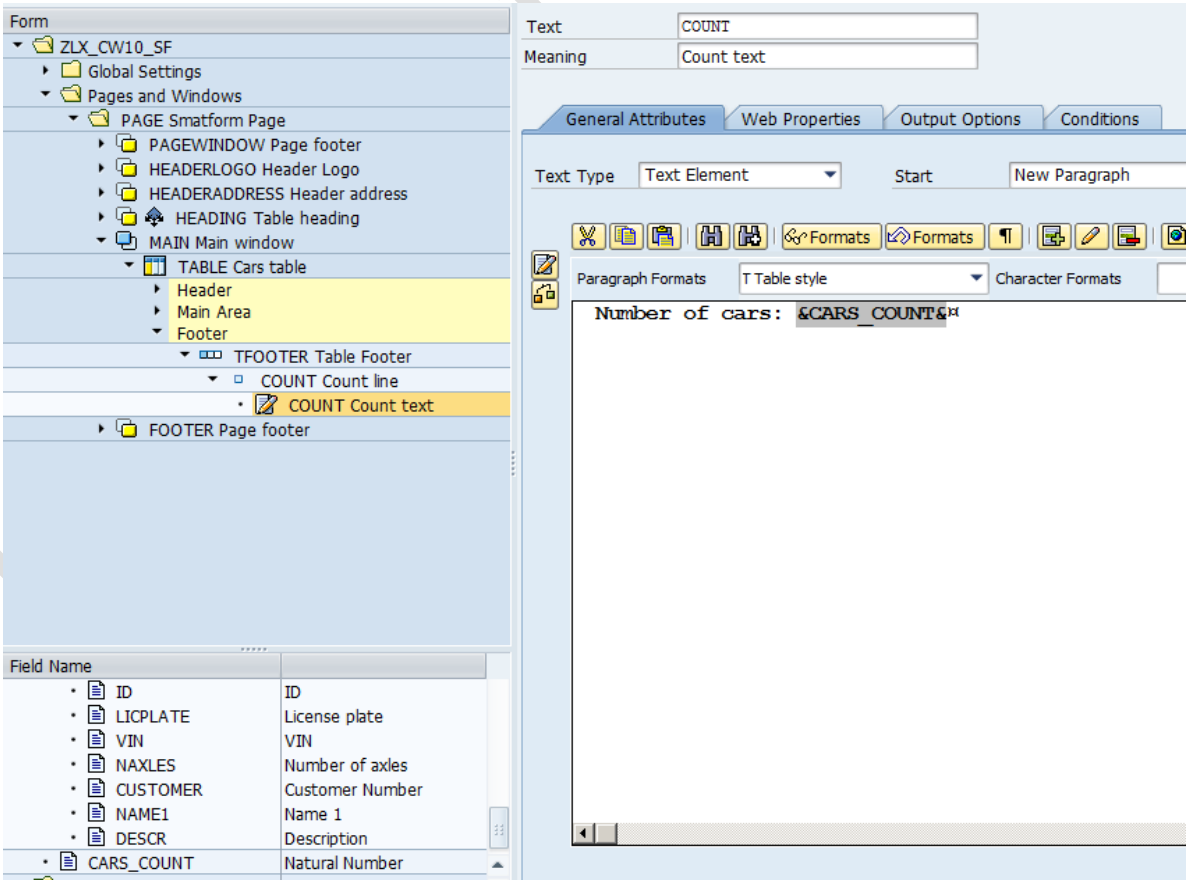
17. Double click on TABLE. Select Details and Insert Row:



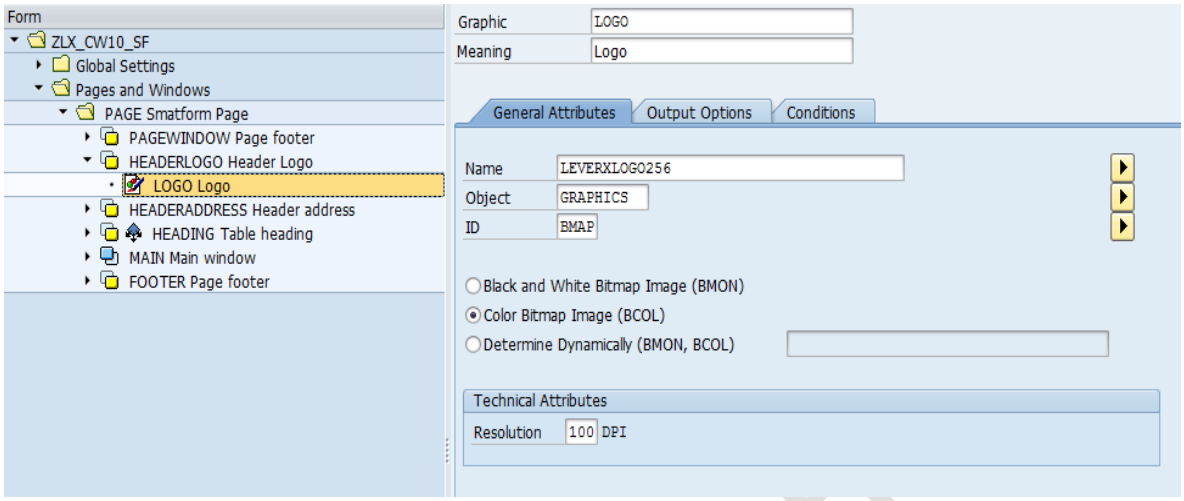
18. Uncheck in Table footer “at Page Break”



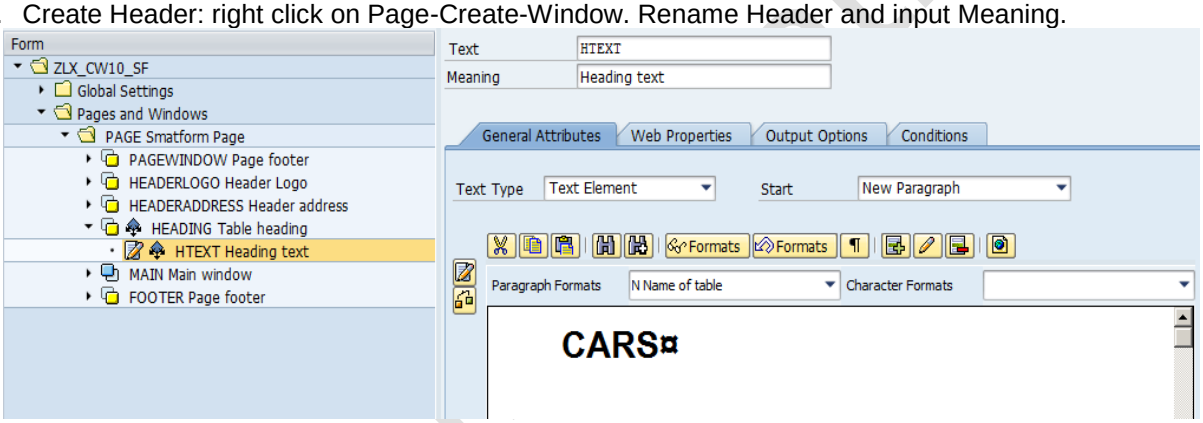
19. Create Table Line in table-> footer. Select appropriate line type. Create Text and add information about cars count.



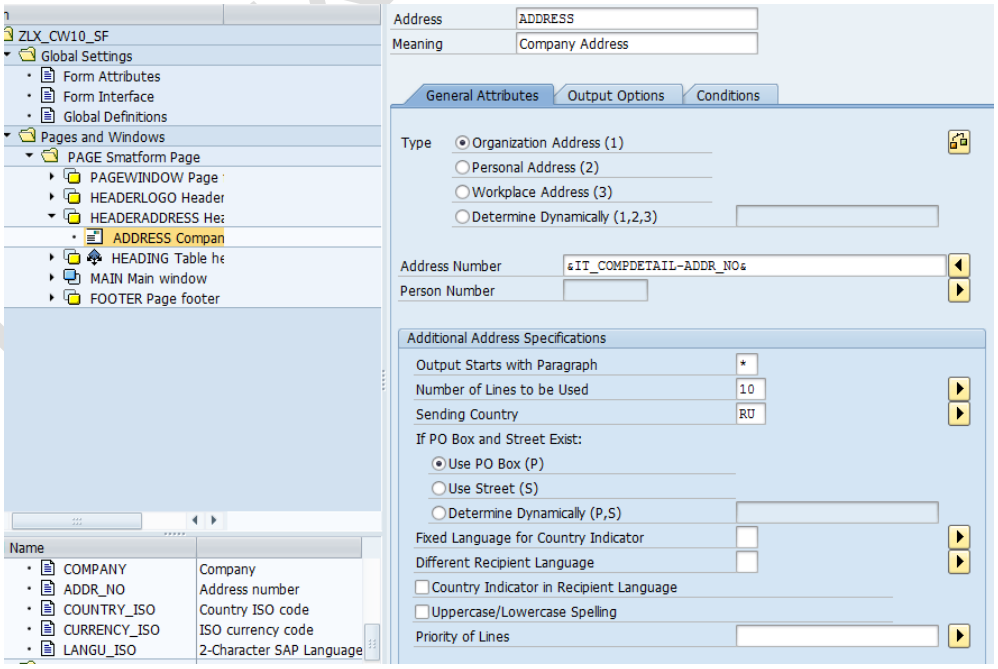
20. Add Logo: right click on Page-Create->Window. Rename Window and add new Meaning. Right click on Window-Create-Graphic. Rename Graphic, add Meaning and input image name:



21. Create Header: right click on Page-Create-Window. Rename Header and input Meaning.

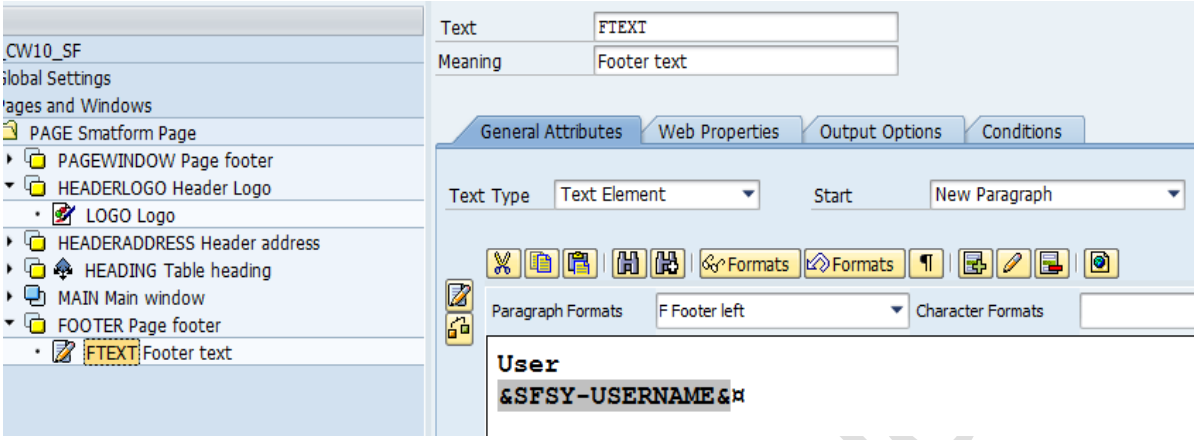


22. Create Address: right click on Header window-Create-Address. Rename Address and input Meaning. Type Address Number from Field Name-Import Interface-IT_COMPDETAIL-ADDR_NO:

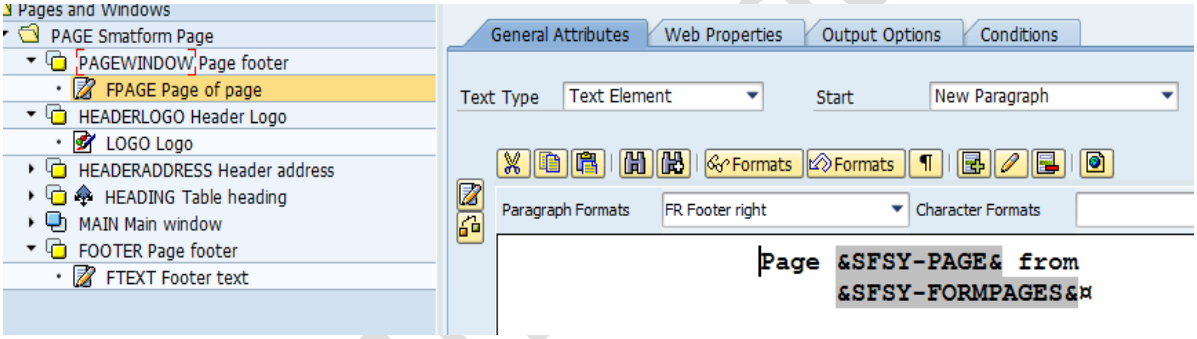


In the report use “BAPI_COMPANYCODE_GETDETAIL” for getting company address features dynamically.

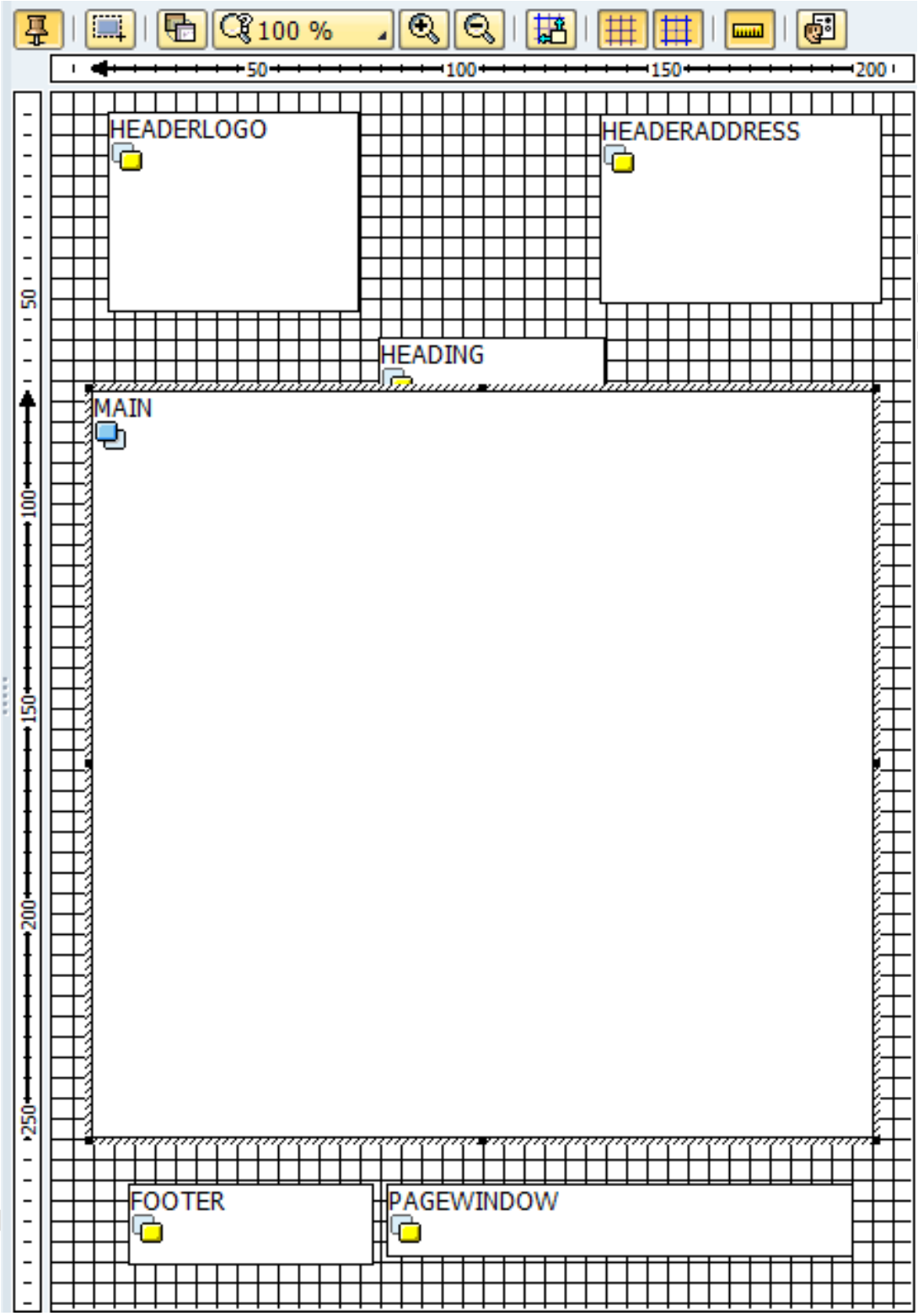
23. Create Footer: Create New Window-Create Text-Add information to the text.



24. Create Window for page numbering.



25. Drag and Drop all elements in Form Painter



26. Create Report for print Smart form.

Company
LeverX
Bogdanovicha 155b
220000 MINSK
BELARUS

CARS

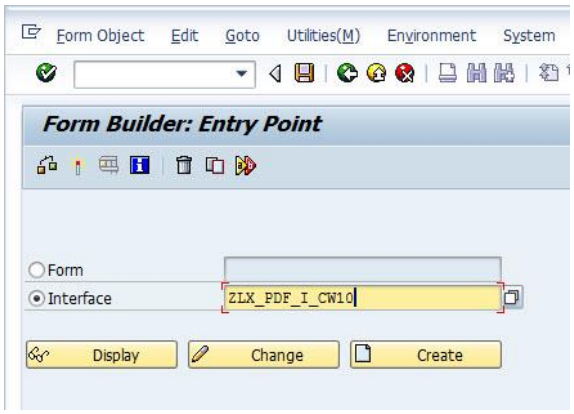
Car ID	License plate	VIN	Number of axles	Description of car	Customer ID	Customer name
1	2344KK-9	5429685374 1858965471	2	LeverX NICE BMW CAR	1	BMW
2	MM56786 1	6846186188 7845554488	3	LeverX BEAUTY WHITE MERSEDES	2	Mercedes
3	32084-22	0938459082 3748962345	4	LeverX PORSCHE 911 TURBO S	3	Porsche
4	NJ3425-54	9085908734 5823495237	3	LeverX PORSCHE PANAMERA	3	Porsche
Number of cars: 4						

User CLEANER

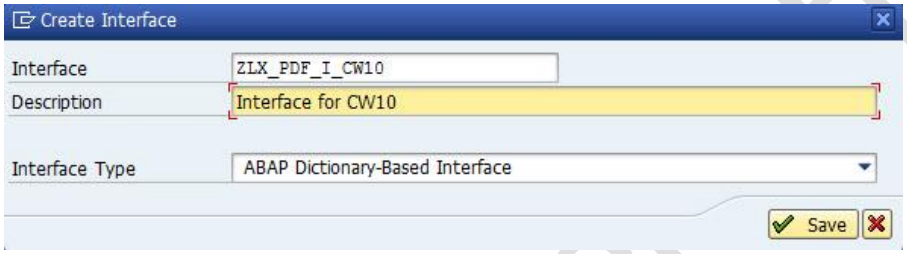
Page 1 from 1

How to create a PDF form

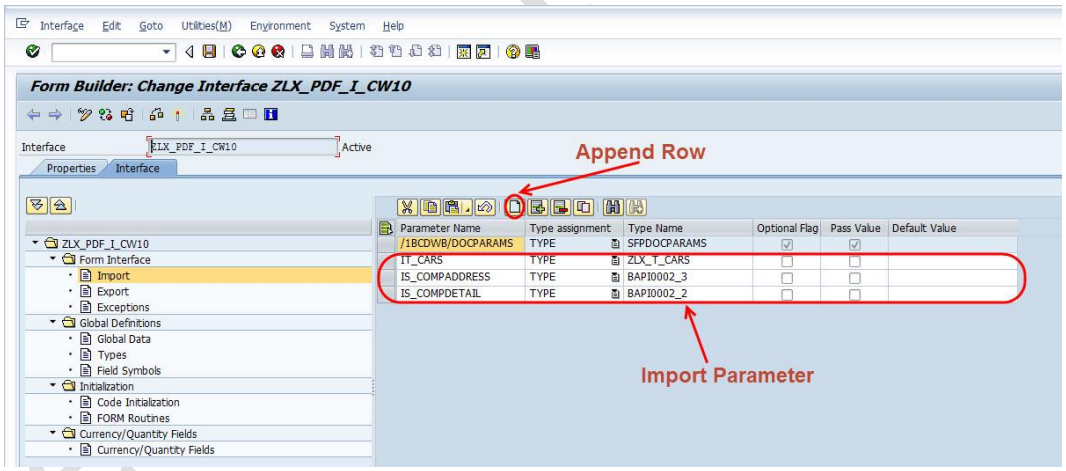
1. Go to transaction SFP to create adobe form. Provide the interface name and click on create button.



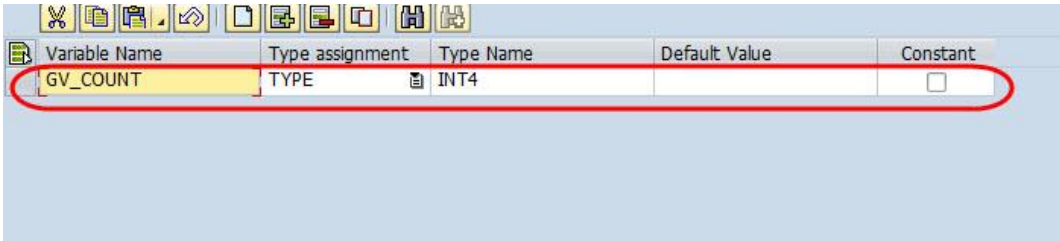
2. Provide the short description and click on Save button.



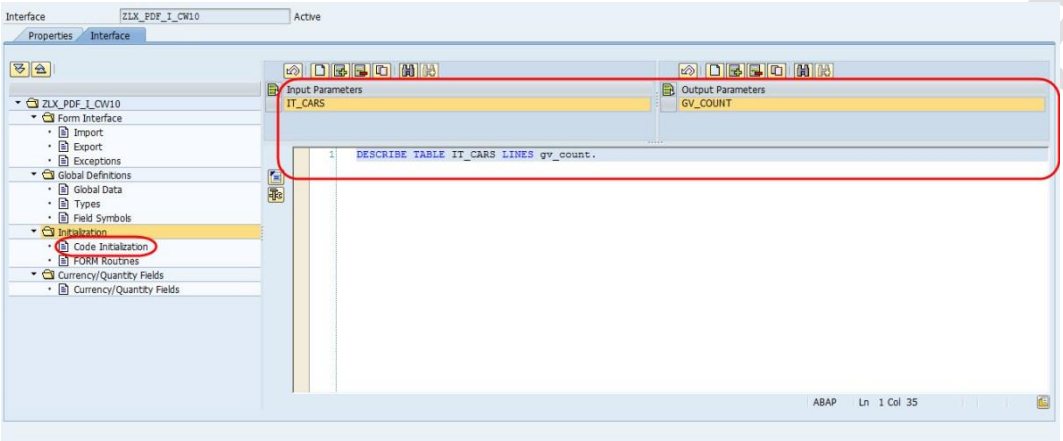
3. Click on Append Row button and enter the parameter.

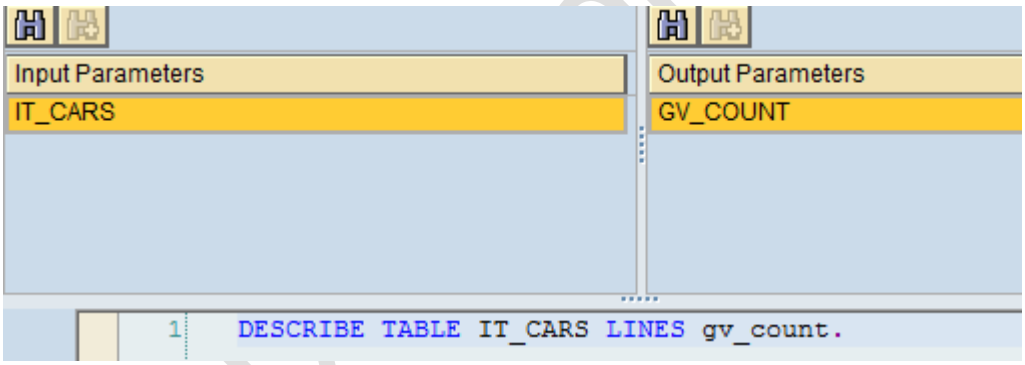


4. Go to Global Data in the Global Definitions and add variable.



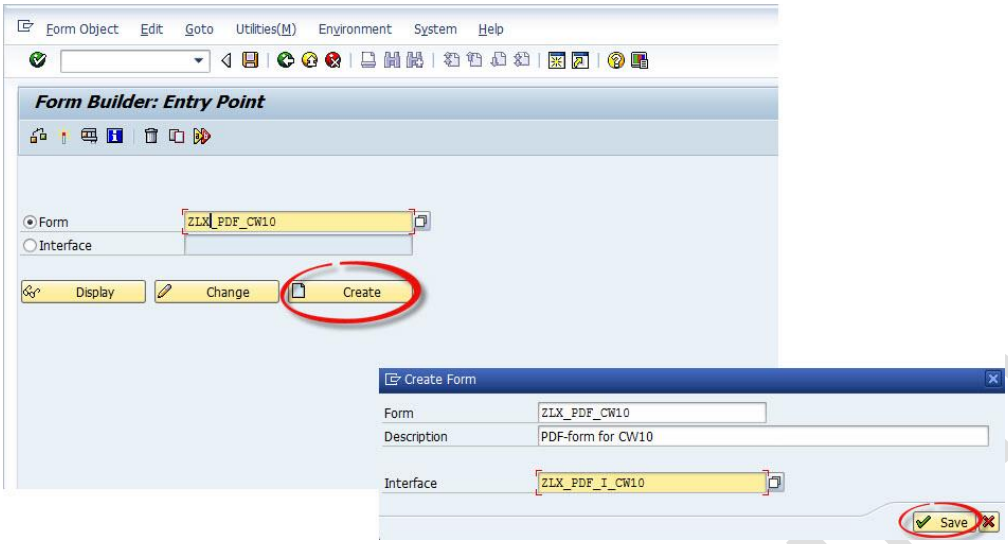
5. Go to Code Initialization in the Initialization, add Input and Output Parameters and Code.



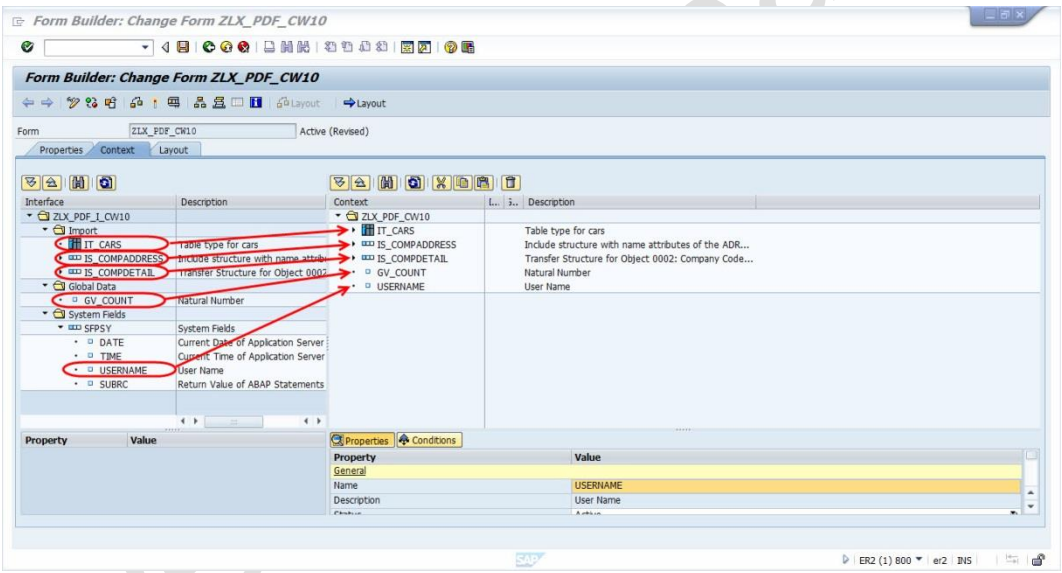


6. Check, Save and Activate.

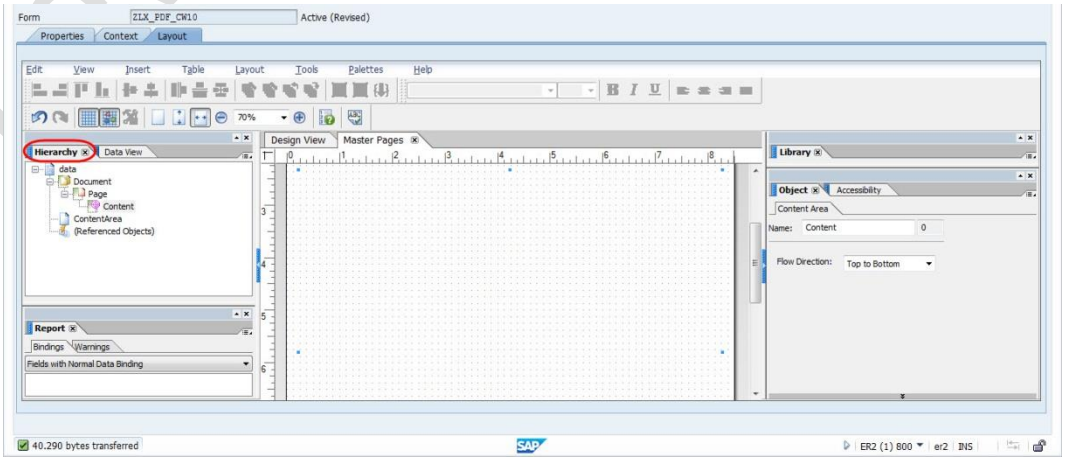
7. Now go back to the initial screen. Provide the form name and click on Create button. Provide the description for the form and interface name created as shown below.



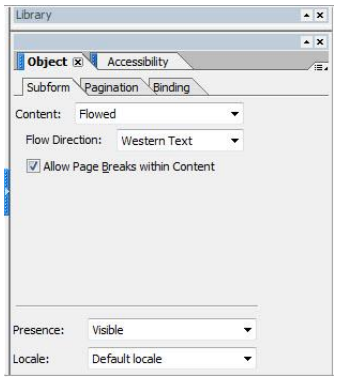
8. Drag and drop the fields to be displayed in the layout from interface to context as shown below.



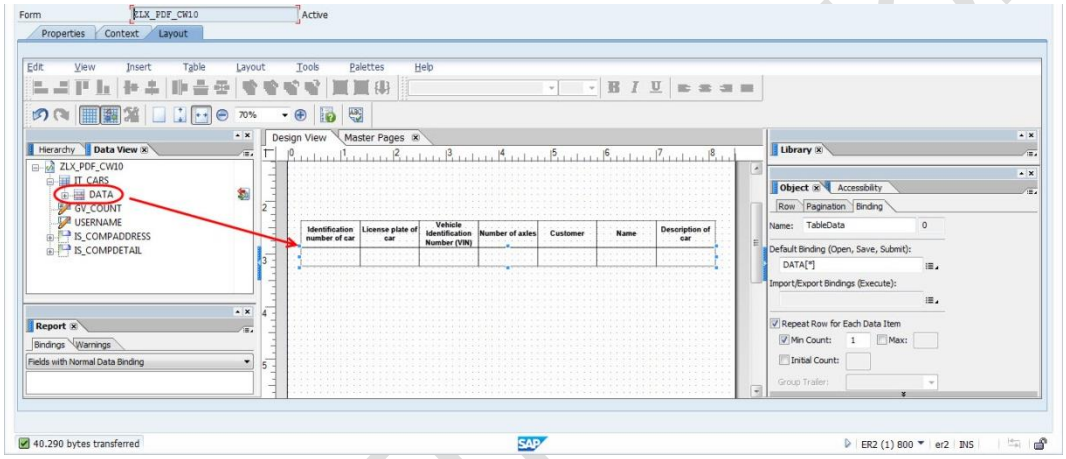
9. Go to layout tab. Select Hierarchy tab. Rename objects: right click-Rename object (F2)



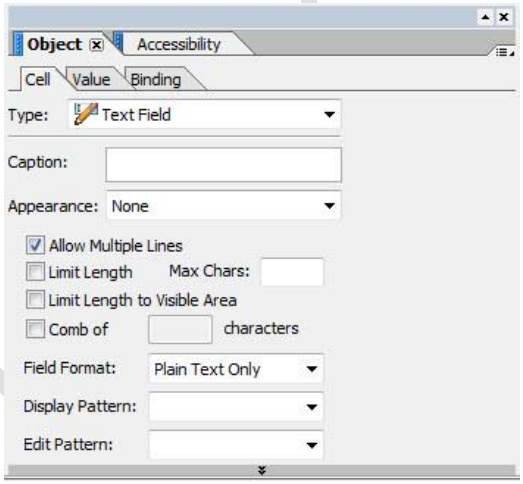
10. Right click on (untitled Subform) and select Insert Subform. Select next parameters for subforms



11. From tab Data View Drag and Drop table on Design View.

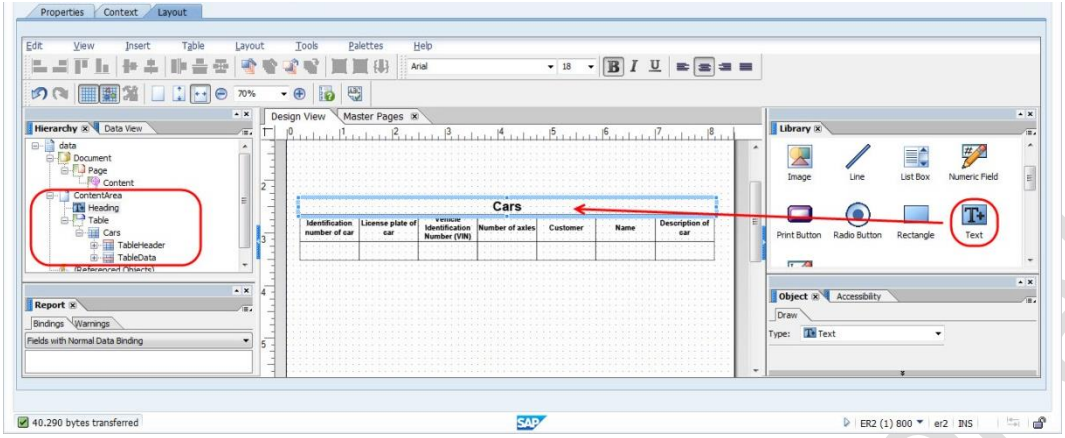


12. For table body cells in object tab set parameter Allow Multiple Lines and delete parameter Limit Length.

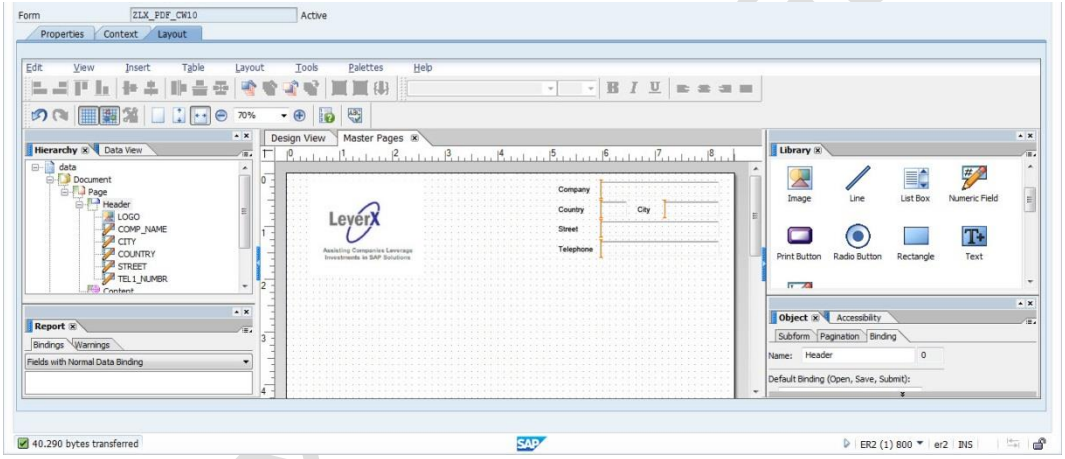


LeverX International, LLC
7A Lopatina str. 3 floor, Minsk, Belarus
Tel +375 17 232-20-01 http://www.leverx.by

13. From Library window Drag and Drop text element on Design View. Text should be only in Subform ContentArea. Rename new element.

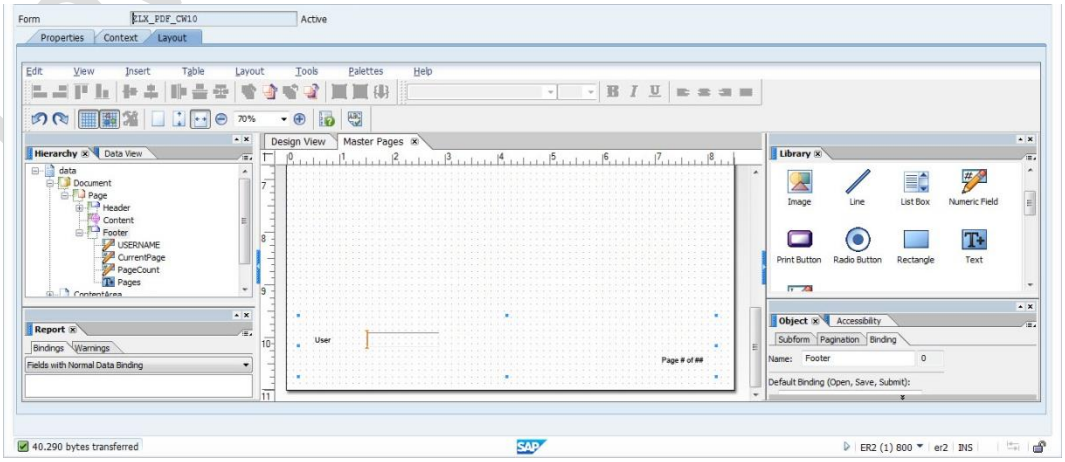


14. In Document-Page insert Subform for header. Add Image element in this Subform from the Library. Select tab Data View and drag and drop necessary address information from IS_COMPADDRESS and IS_COMPDETAIL. In tab Hierarchy rename new objects if necessary.

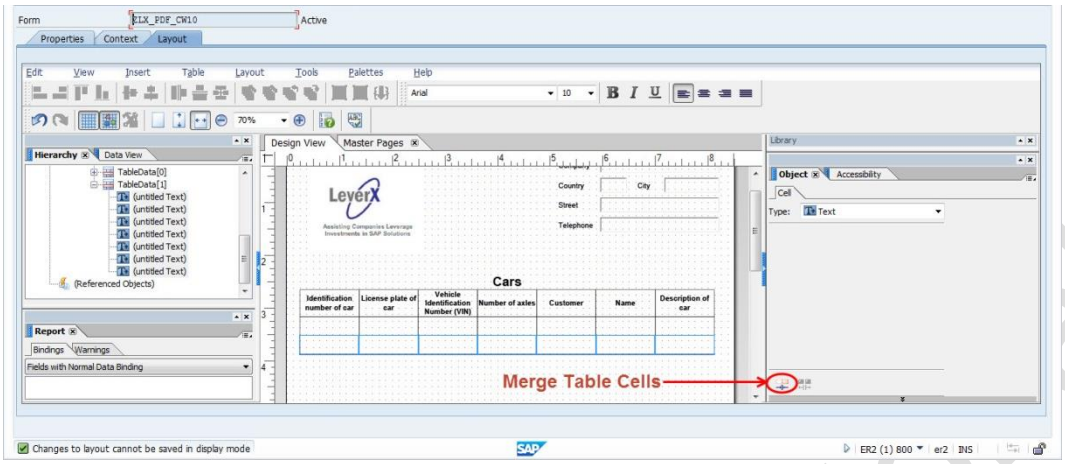


In the report use “BAPI_COMPANYCODE_GETDETAIL” for getting company address features dynamically.

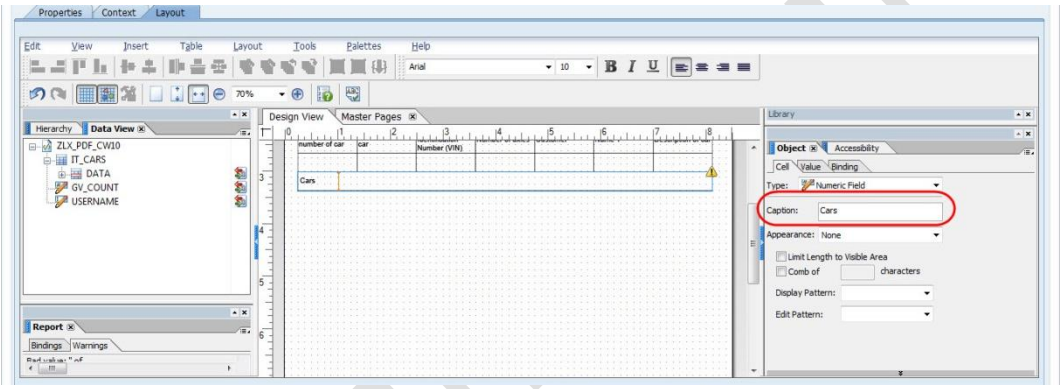
15. In Document->Page insert Subform for footer. Drag and Drop Subform in Page footer. Add Username from Data View and element “Page n of m” from Library. Rename new element if necessary.



16. In Hierarchy in table right click on last row and select Insert Row Below. Select all cells and merge table cells. Rename elements if necessary.



17. Drag and Drop GV_COUNT from Data View into footer table row. Edit Caption in Object tab.



18. Save and Activate form.

19. Create report.

Assisting Companies Leverage Investments in SAP Solutions

Company	LeverX International		
Country	BY	City	Minsk
Street	Bogdanovicha		
Telephone	7772233		

Cars

Identification number of car	License plate of car	Vehicle Identification Number (VIN)	Number of axles	Customer	Name	Description of car
1	7865KK-7	23434434123245342223	2	ADEF	Asstra DE Express	NEW
2	2344KK-2	435345354245 4323443	2	LUKOIL	Lukoil	BROKEN
3	MM5678	RT345344RE3222234675	3	MAGAZIN01	ЗАО Магазин номер 1	GOOD CAR
4	7965KK-5	55345345343453453455	4	10	Asstra Belarus	NEW
5	0001MP-05	RT345344RE3222234675	3	MAGAZIN01	ЗАО Магазин номер 1	BROKEN
6	0001MP-06	55345345343453453455	4	10	Asstra Belarus	GOOD CAR
7	2001MP-05	23434434123245342223	2	ADEF	Asstra DE Express	NEW
8	201MP-06	435345354245 4323443	2	LUKOIL	Lukoil	NEW
9	0001MP-07	RT345344RE3222234675	3	MAGAZIN01	ЗАО Магазин номер 1	BROKEN
10	0001MP-08	55345345343453453455	4	ABTB	Asstra BY Transport Brest	BROKEN
11	0061MP-07	55345344563453453455	2	ABTM	Asstra BY Transport Minsk	NEW
12	2001MP-08	55345345343433333333	2	ABTG	Asstra BY Transport	GOOD CAR
13	0001MP-09	23434434123245342223	2	ADEF	Asstra DE Express	NEW
14	0001MP-10	435345354245 4323443	2	LUKOIL	Lukoil	GOOD CAR
15	0002MP-01	RT345344RE3222234675	3	MAGAZIN01	ЗАО Магазин номер 1	NEW
16	0003MP-02	55345345343453453455	4	10	Asstra Belarus	NEW
17	2001MP-05	23434434123245342223	2	ADEF	Asstra DE Express	GOOD CAR

User username

Page 1 of 2

LeverX International, LLC
7A Lopatina str. 3 floor, Minsk, Belarus
Tel +375 17 232-20-01 http://www.leverx.by

Appendix 1 - Smart Forms Parameters of the Generated Function Module

From the function module's point of view there are import (K= I) and export parameters (K= E)

PARAMETER NAME	K	TYPE	DESCRIPTION
ARCHIVE_INDEX	I	TOA_DARA	SAP ArchiveLink parameter for archiving forms
ARCHIVE_INDEX_TAB	I	TSFDARA	
ARCHIVE_PARAMETERS	I	ARC_PARAMS	
CONTROL_PARAMETERS	I	SSFCTRLOP	Control structure for the general control of the form output (output medium, output with/without dialog, language, and so on)
MAIL_APPL_OBJ	I	SWOTOBJID	Business Communication Interface (BCI) parameter for sending forms as e-mail
MAIL_RECIPIENT	I	SWOTOBJID	
MAIL_SENDER	I	SWOTOBJID	
OUTPUT_OPTIONS	I	SSFCOMPOP	Structure with output options for output to spool, as FAX, or in XSF format
USER_SETTINGS	I	TDBOOL	If the parameter is set ('X'), SAP Smart Forms copies the user defaults for the Spool Control, which you set under System User -> Profile Own Data (Output Device, Print Immediately, Delete after Output). If it is not set, SAP Smart Forms instead evaluates the following parameters of the structure for the output options (SSFCOMOP): Printer settings (TDDEST, TDPRINTER, RQPOSNAME) TDIMMED (Print immediately) TDDELETE (Delete after output)
DOCUMENT_OUTPUT_INFO	E	SSFCRESPD	This structure contains nothing but the TDFPAGES field of type TDFPAGES , which contains the number of printed form pages
JOB_OUTPUT_INFO	E	SSFCRESCL	After form output you can use this structure to query which actions produced which results
JOB_OUTPUT_OPTIONS	E	SSFCRESOP	This structure contains a subset of the fields of OUTPUT_OPTIONS : the fields that the user is allowed to change. It enables you to determine whether your settings have been modified.

LeverX International, LLC
7A Lopatina str. 3 floor, Minsk, Belarus
Tel +375 17 232-20-01 http://www.leverx.by

Appendix 2 - PDF Parameters of the Generated Function Module

Parameters of the Structure SFPDOCPARAMS (/1BCDWB/DOCPARAMS)

PARAMETER NAME	TYPE	MEANING
LANGU	LANGU	Language in which the form is displayed
REPLANGU1	LANGU	If the form does not exist in the language specified in LANGU, the system evaluates these fields in the given sequence instead.
REPLANGU2	LANGU	
REPLANGU3	LANGU	
COUNTRY	LAND1	Country key for date and number formatting. No setting is required if the SET COUNTRY command is used in the application program before the generated function module is called, and the same country is specified there.
FILLABLE	FPINTERACTIVE	FILLABLE = X generates an interactive form. This form can be displayed and edited in Adobe Acrobat or Adobe Reader. Usage rights are allocated to the form; these rights enable data to be entered and saved in Adobe Reader. FILLABLE = N generates an interactive form without usage rights.
DYNAMIC	FPDYNAMIC	DYNAMIC = X generates an interactive, dynamic form to be displayed and edited in Adobe Acrobat or Adobe Reader versions higher than 7.0. This parameter is valid only if the FILLABLE parameter is also set.
DARATAB	TFPDARA	If you want to archive the generated form, you must make at least one entry in this table (with archiving indexes).

Parameters of the Structure FPFORMOUTPUT (/1BCDWB/FORMOUTPUT)

PARAMETER NAME	TYPE	MEANING
PDF	FPCONTENT	Contains the generated PDF. A PDF is generated only on demand by the GETPDF parameter of the function module FP_JOB_OPEN in the application program, or by the print preview function.
PDL	LANGU	Contains the generated PDL.
PAGES	LANGU	Contains the number of generated pages in the form.
LANGU	LANGU	Language key