

Lection 5 Workbook

Extended development

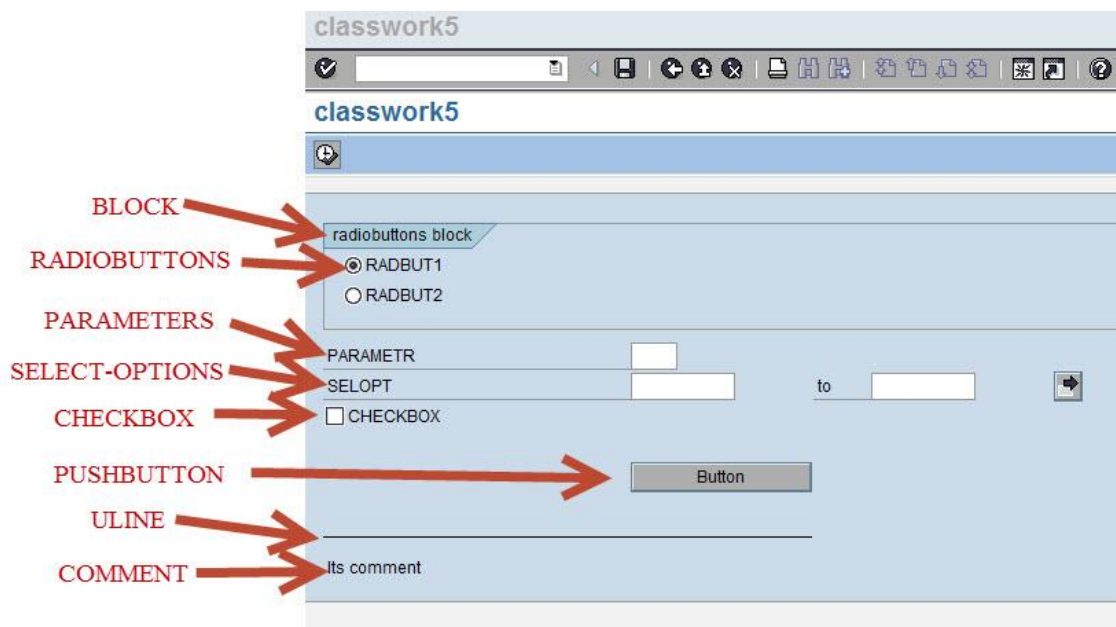
Contents

How to create a selection screen.....	2
How to create a custom screen.....	3
How to create a BAL LOG.....	11
How to create a Message class.....	14
How to create a GUI-status.....	16

How to create a selection screen

You often use screens purely for data input. In these cases, you can use a selection screen. Selection screens provide a standardized user interface in the R/3 System. Users can enter both single values and complex selections. Input parameters are primarily used to control the program flow, while users can enter selection criteria to restrict the amount of data read from the database. You can create your own selection screen.

You can use: PARAMETERS, SELECT-OPTIONS, RADIOBUTTON, CHECKBOX, PUSHBUTTON, ULINE, COMMENT keywords. Also you can assign special name for any element.



For creating these elements you should use:

RADIOBUTTONS

PARAMETERS: <name1> RADIOBUTTON GROUP <grp_name1>,
<name2> RADIOBUTTON GROUP <grp_name1>.

BLOCK

SELECTION-SCREEN BEGIN OF BLOCK <block_name1> **WITH FRAME TITLE** text-t01.

...

SELECTION-SCREEN END OF BLOCK <block_name1>.

PARAMETERS

PARAMETERS: <name> **LIKE** <type>.

SELECT-OPTIONS

SELECT-OPTIONS: <name> **FOR** <type>.

PUSHBUTTON

SELECTION-SCREEN PUSHBUTTON <name>.

ULINE

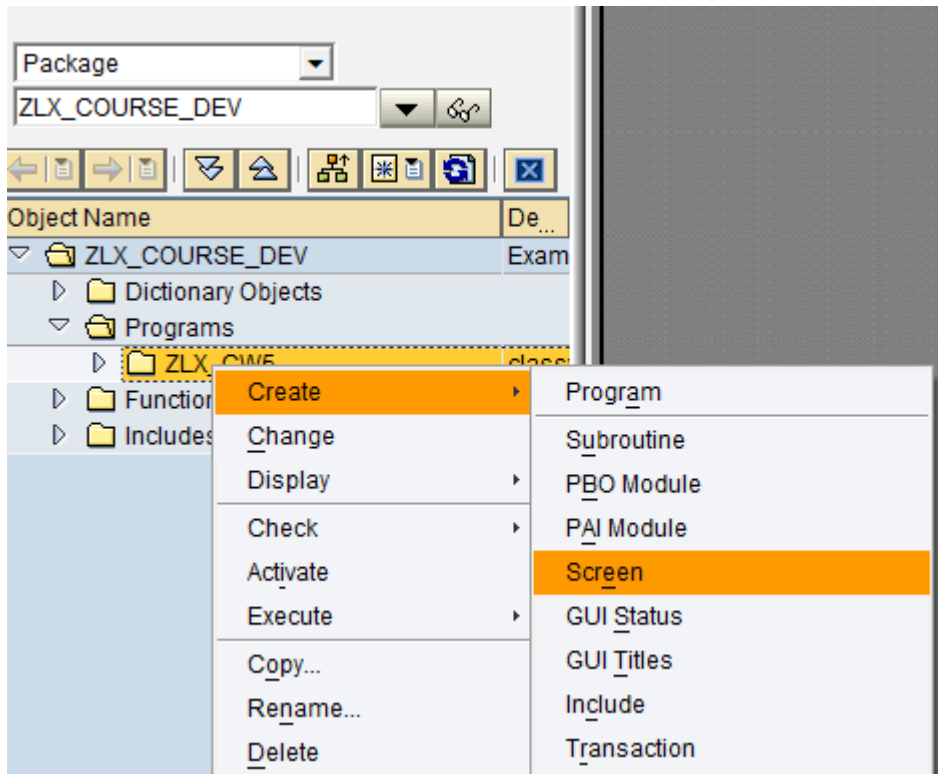
SELECTION-SCREEN ULINE /1(50).

COMMENT

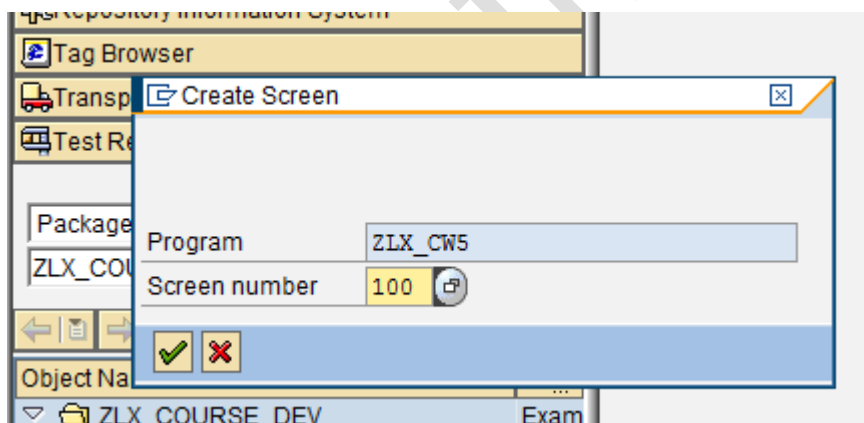
SELECTION-SCREEN COMMENT /1(20) <var_name>.

How to create a custom screen

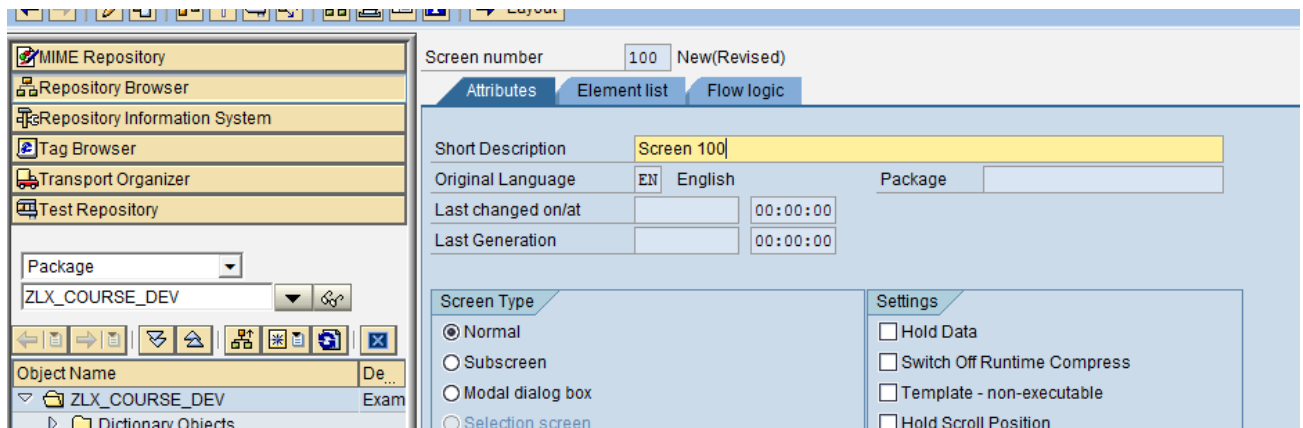
Go to transaction SE80.
Select your package and select your program.
Right click on your program, choose Create → Screen



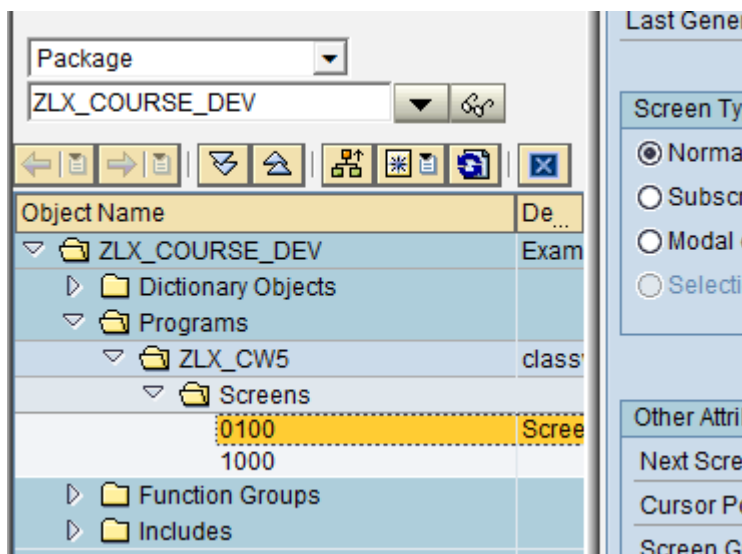
Provide the screen number



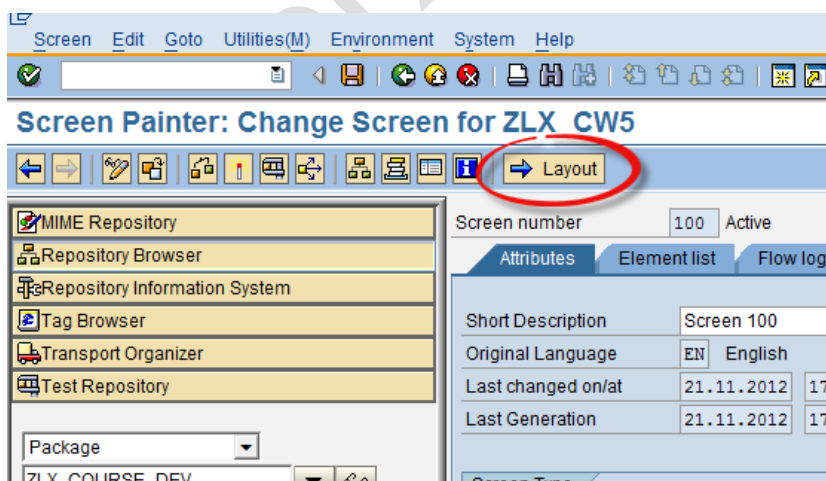
Provide the short description



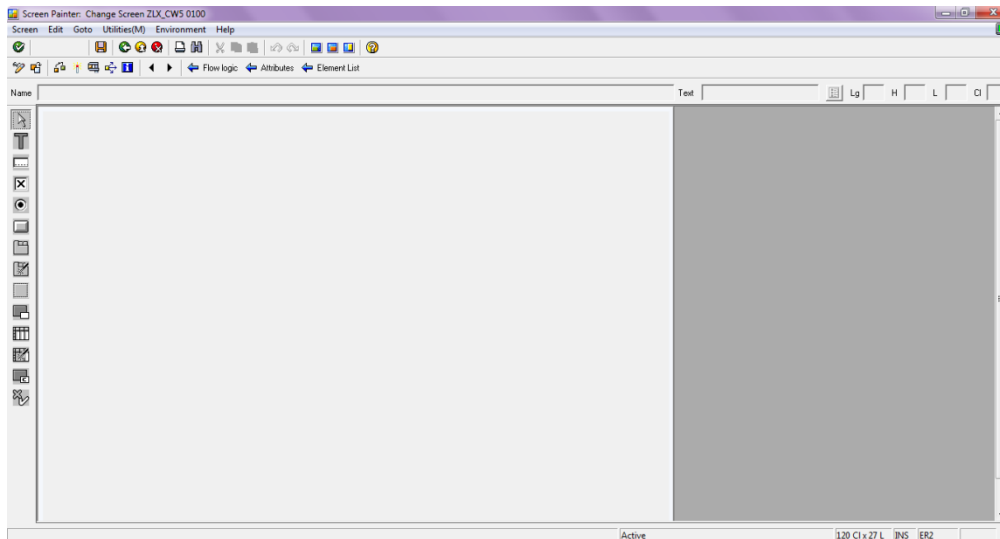
Save it and activate it.
Now you can see your screen



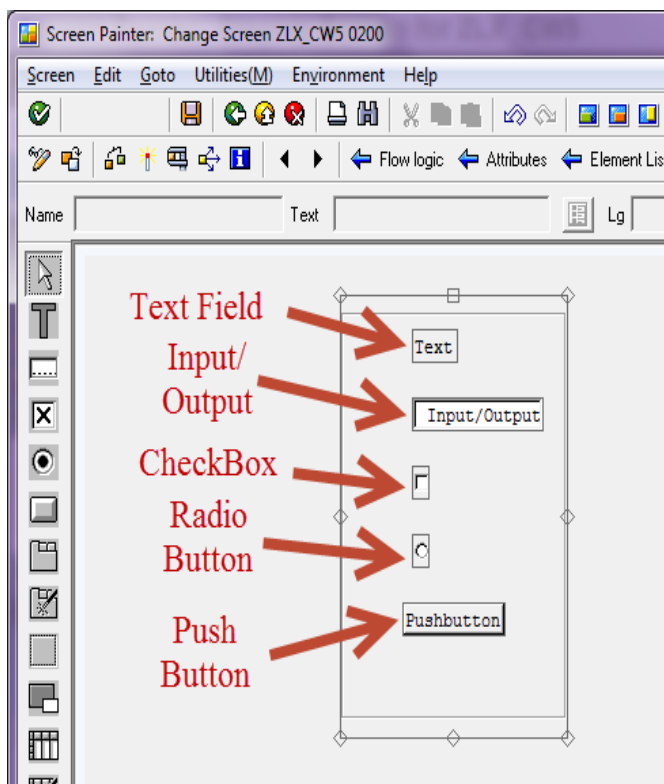
Click Layout button to design screen.



Now you can see a screen designer



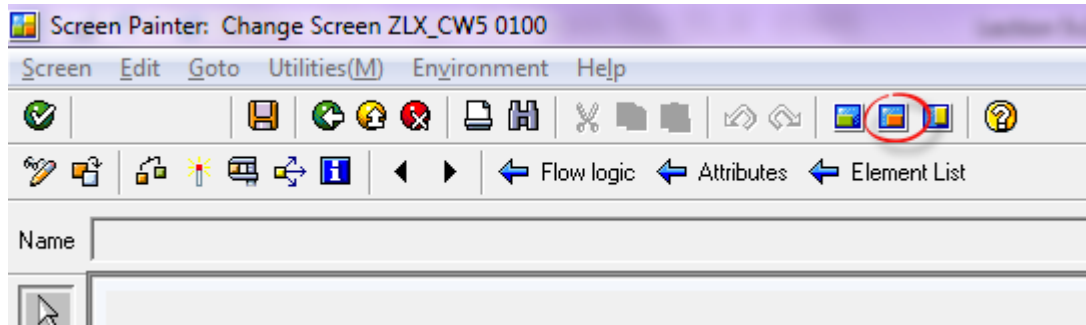
You can drag-and-drop elements from toolbar.



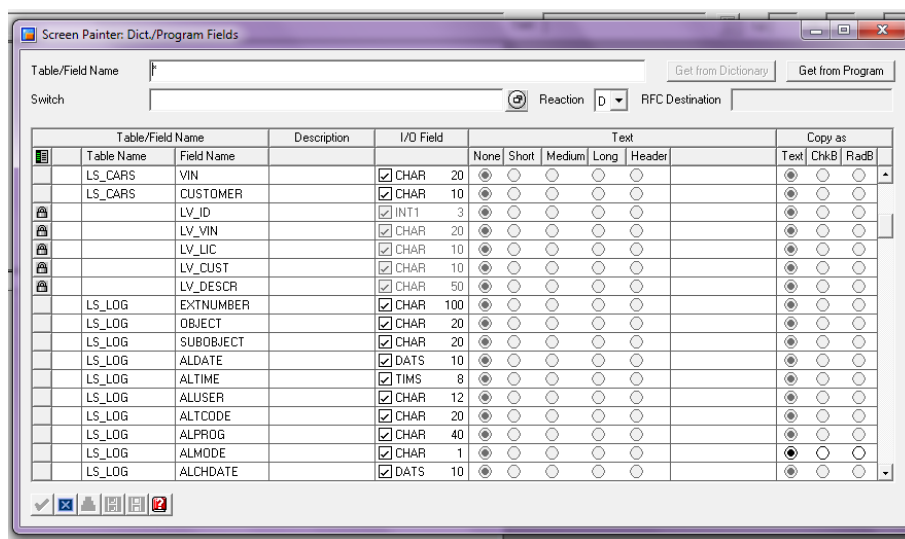
If you want to display variable from your report on the screen you need:

1. Create variables in source code.
DATA: lv_id TYPE zlx_cars_id,
 lv_vin TYPE zlx_cars_vin,
 lv_lic TYPE zlx_cars_licplate,
 lv_cust TYPE kunnr,
 lv_descr TYPE zlx_cars_desc.

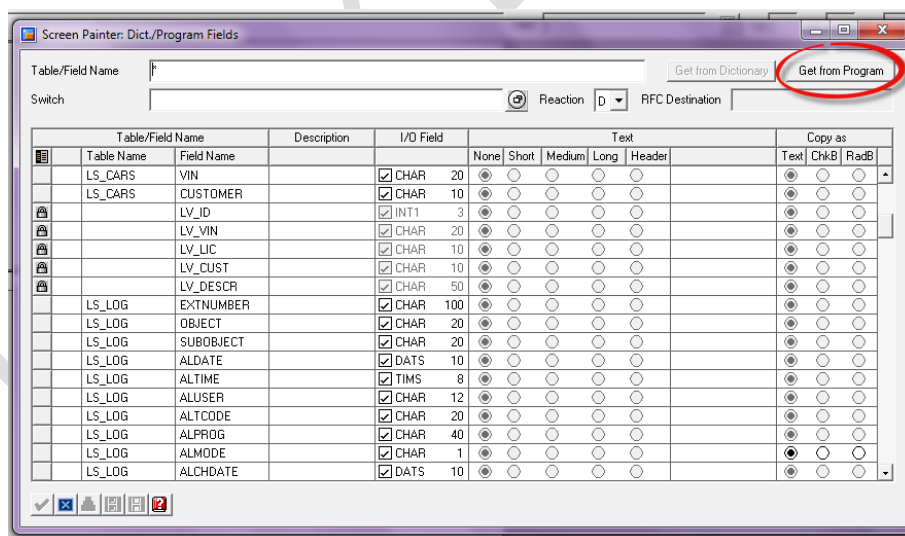
- Open screen designer (button Layout). Press button Dictionary/Program fields window.



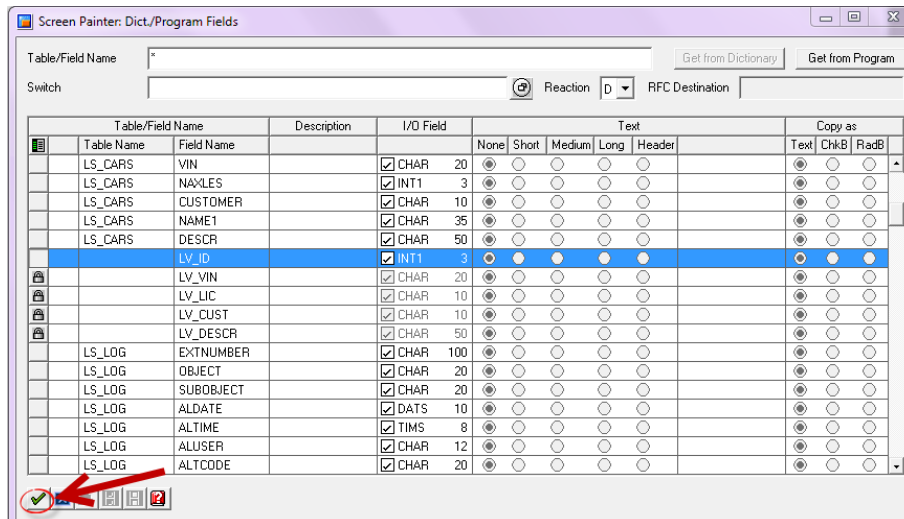
- Now you can see Screen painter window



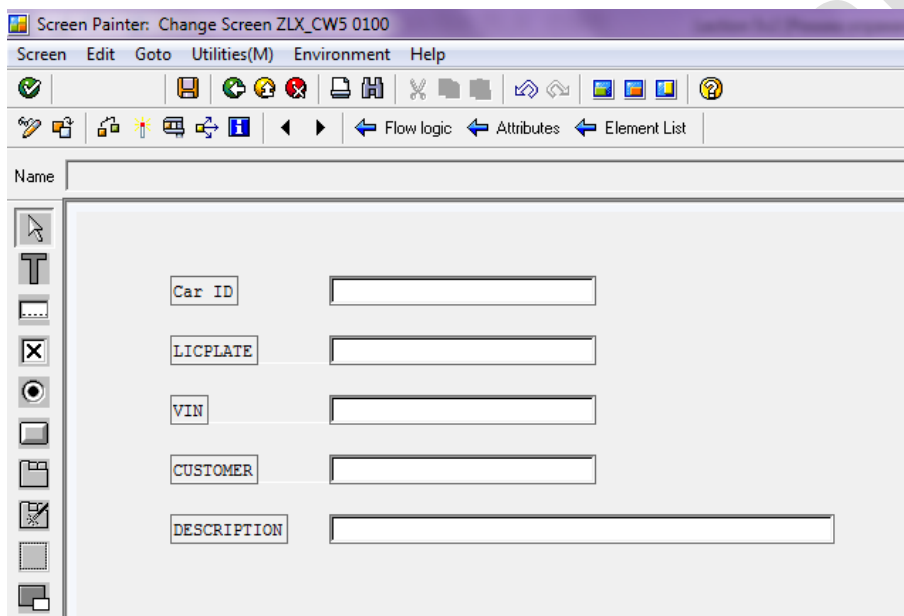
- Press button Get from Program



5. Select row with necessary parameter and press Ok button

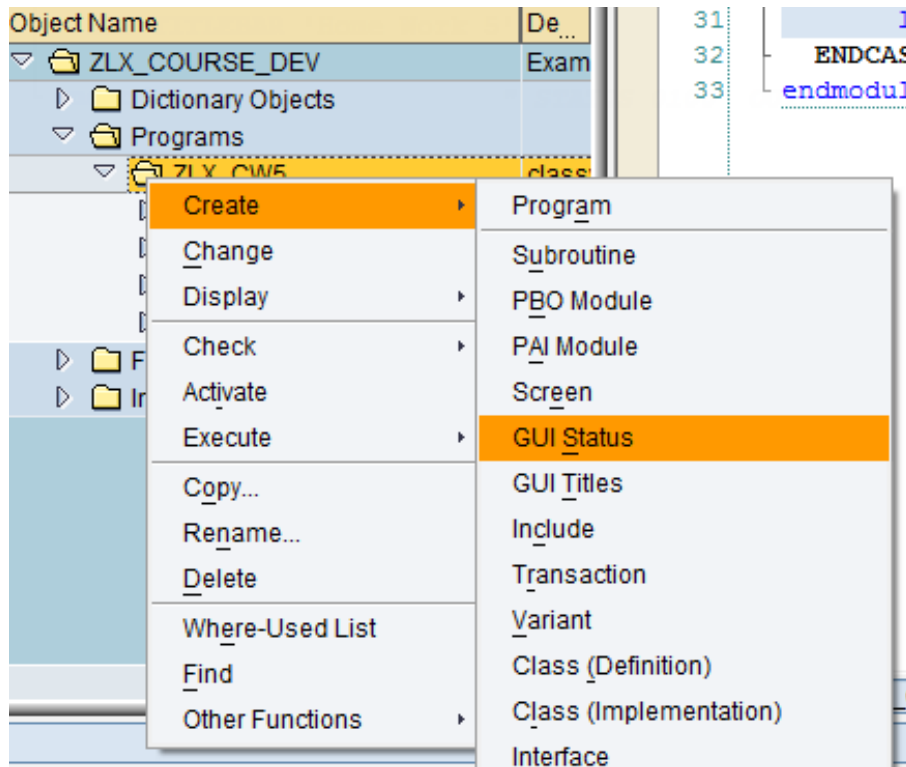


6. Continue adding other parameters.

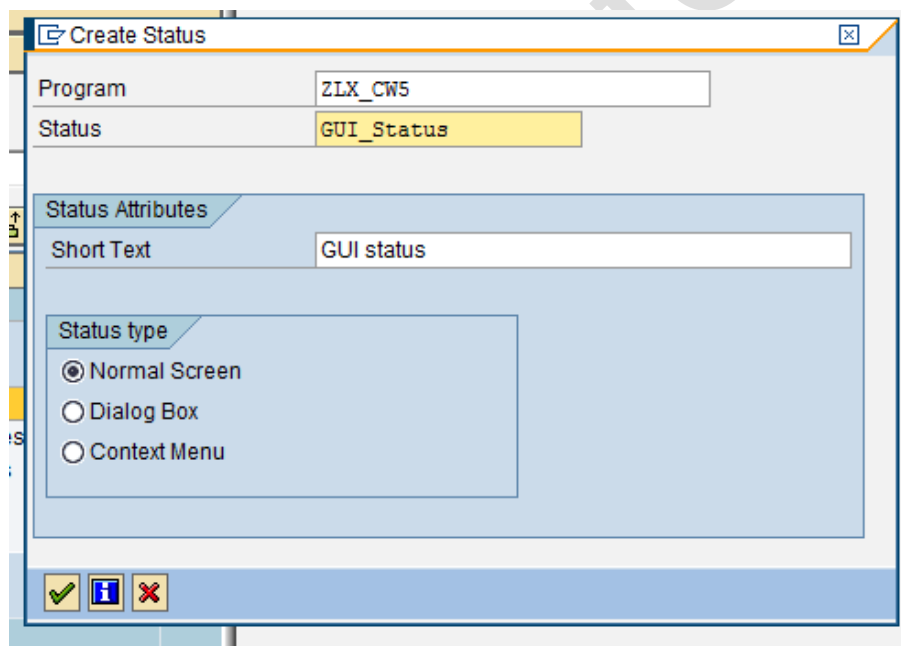


Save it and activate it.

Now you need to create GUI Status. Go to transaction SE80. Click on your program with right mouse-button, choose Create→ GUI Status.

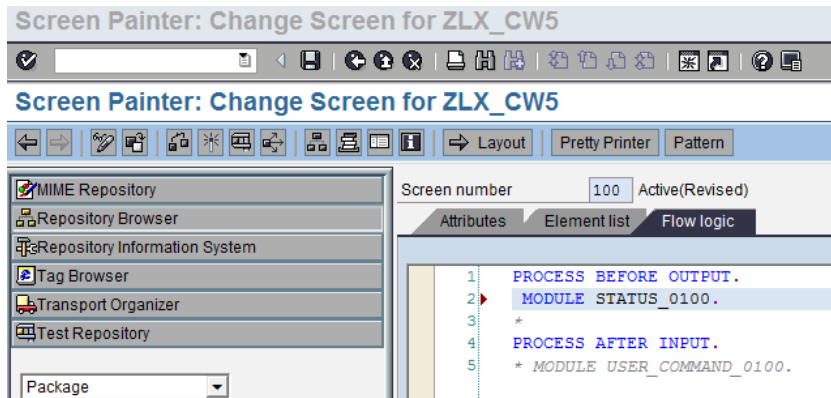


Provide the status name and the short text.

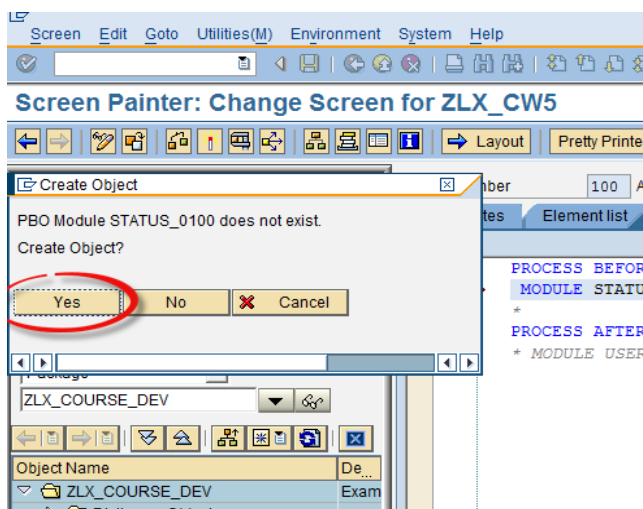


You can see two events: PBO(PROCESS BEFORE OUTPUT) and PAI(PROCESS AFTER INPUT).

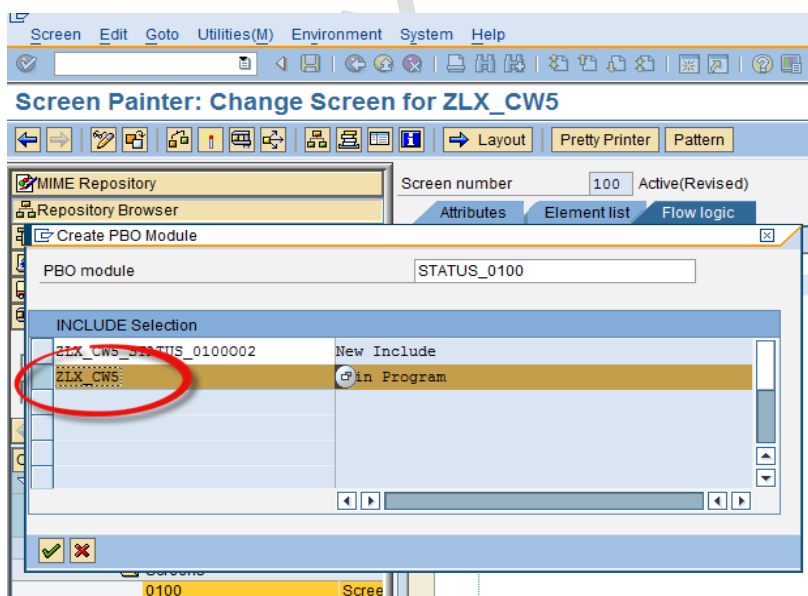
Uncomment string **MODULE STATUS_0100**.



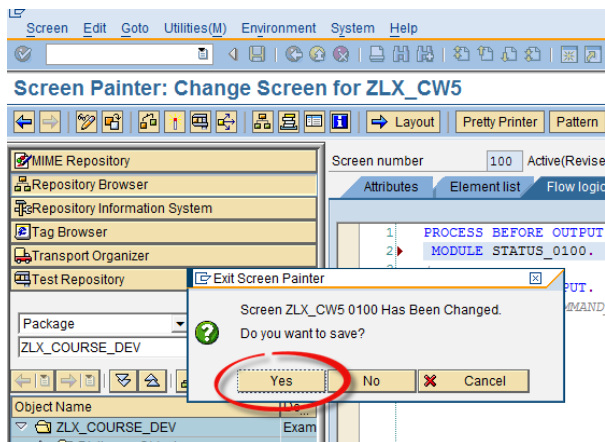
Double click on it. Choose Yes.



Select in Program.



Click Yes



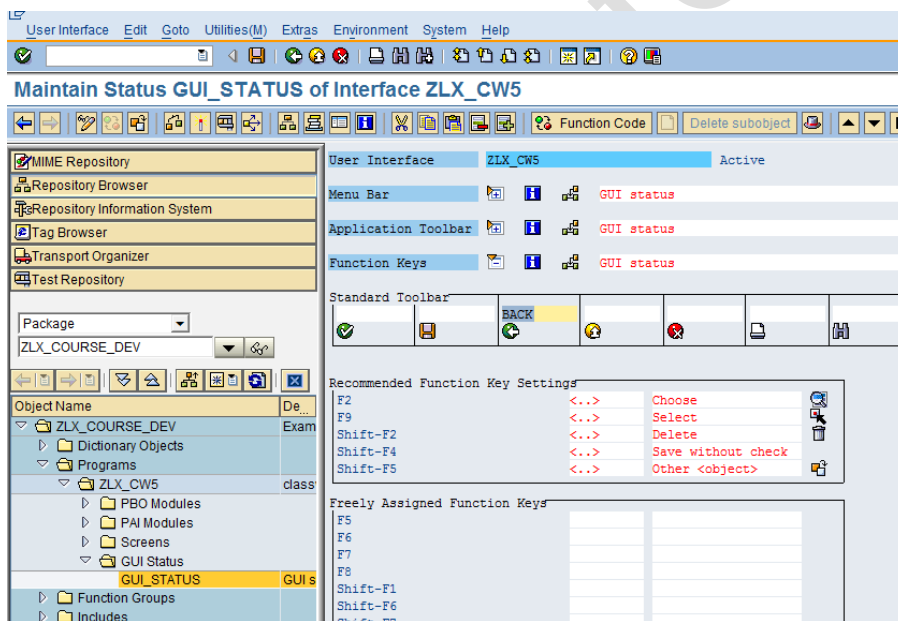
System automatically creates source code.

Uncomment lines `SET PF-STATUS 'xxxxxxx'` and `SET TITLEBAR 'xxx'`.
Enter PF-STATUS 'GUI_STATUS' and TITLEBAR 'Class Work 5'.

```
module STATUS_0100 output.  
  SET PF-STATUS 'GUI_STATUS'.  
  SET TITLEBAR 'Class Work 5'.
```

Save it & activate it.

Now you should create BACK button in GUI status. Choose GUI_STATUS → Function Keys. Enter name for button BACK.



Go to Screens, choose your screen. Uncomment line: `MODULE USER_COMMAND_0100` and double click on it. System automatically creates source code. Enter this code:

```
CASE sy-ucomm.  
  WHEN 'BACK'.  
    leave to screen 0.  
ENDCASE.
```

Save it and activate it.

How to create a BAL LOG

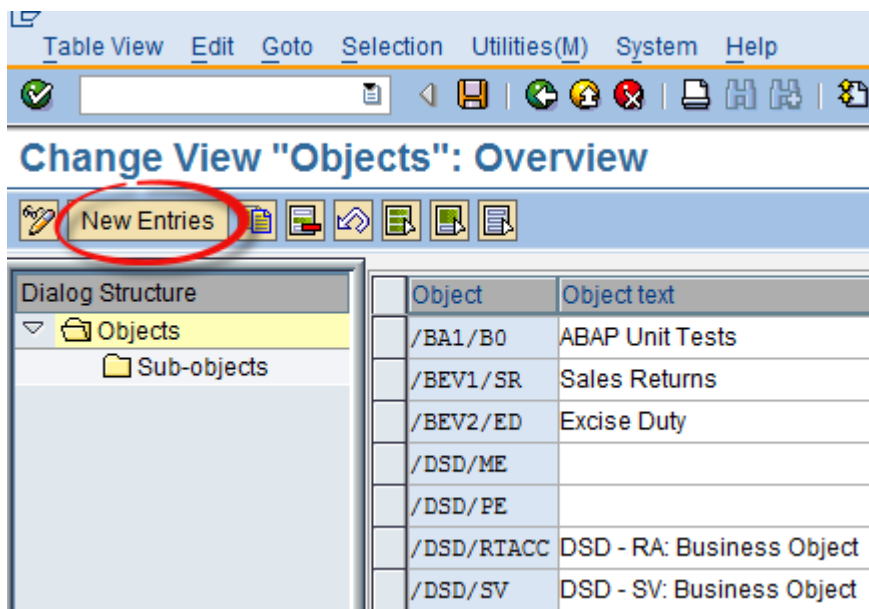
Application Log provides an infrastructure for collecting messages and exceptions in a log, saving, reading and deleting logs in the database and displaying them.

How to use Application Log

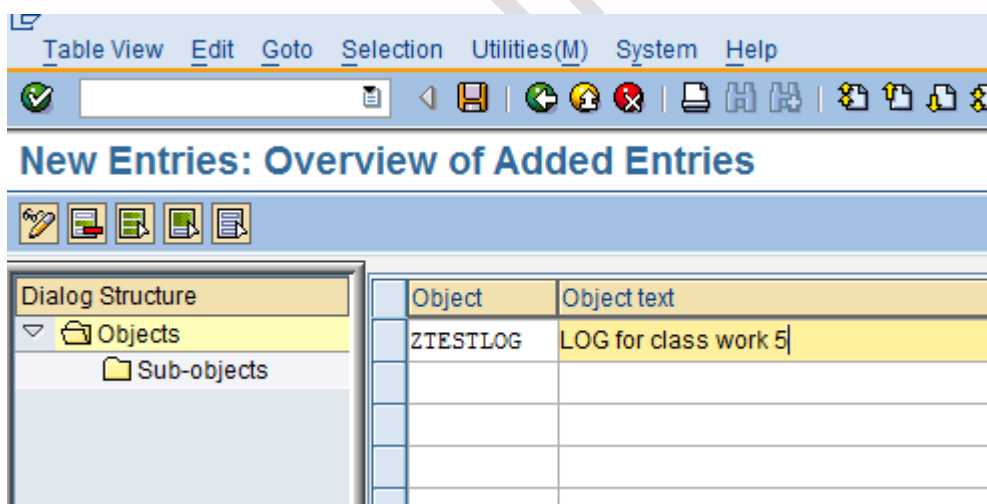
There are different transactions for use of the Application Log.

- ☐ Developer: Use transaction SLG0 to define entries for your own applications in the application log.
- ☐ Key user: Use transaction SLG1 to analyse the application log.
- ☐ Administrator: Use transaction SLG2 to delete logs.

Go to transaction SLG0. Press button New Entries.



Provide the name of the objects and Object text. Save it.



We need to add code to create LOG, add messages to LOG, save LOG and display LOG.

Create LOG:

```
DATA: ls_log TYPE bal_s_log,
      gv_log_handle TYPE balloghndl,
      ls_display_profile TYPE bal_s_prof,
      ls_msg TYPE bal_s_msg.
ls_log-extnumber = 'Application Log for Class Work 5'.
ls_log-object = 'ZTESTLOG'.
ls_log-aldate = sy-datum.
ls_log-altime = sy-uzeit.
ls_log-aluser = sy-uname.
ls_log-alprog = sy-repid.
CALL FUNCTION 'BAL_LOG_CREATE'
  EXPORTING
    i_s_log = ls_log
  IMPORTING
    e_log_handle = gv_log_handle
  EXCEPTIONS
    log_header_inconsistent = 1
    OTHERS = 2.
IF sy-subrc <> 0.
  MESSAGE ID sy-msgid TYPE sy-msgty NUMBER sy-msgno
    WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
ENDIF.
```

Add message:

```
ls_msg-msgty = sy-msgty.
ls_msg-msgid = sy-msgid.
ls_msg-msgno = sy-msgno.
ls_msg-msgv1 = sy-msgv1.
ls_msg-msgv2 = sy-msgv2.
ls_msg-msgv3 = sy-msgv3.
ls_msg-msgv4 = sy-msgv4.
```

```
CALL FUNCTION 'BAL_LOG_MSG_ADD'
  EXPORTING
    i_log_handle = gv_log_handle
    i_s_msg = ls_msg
  EXCEPTIONS
    log_not_found = 1
    msg_inconsistent = 2
    log_is_full = 3
    OTHERS = 4.
```

Save LOG:

```
CALL FUNCTION 'BAL_DB_SAVE'
  EXPORTING
    i_save_all = 'X'
  EXCEPTIONS
    OTHERS = 1.
IF sy-subrc <> 0.
  MESSAGE ID sy-msgid TYPE sy-msgty NUMBER sy-msgno
    WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
ENDIF.
```

Show LOG:

CALL FUNCTION 'BAL_DSP_PROFILE_POPUP_GET'
EXPORTING

start_col = 5
start_row = 5
end_col = 87
end_row = 25

IMPORTING

e_s_display_profile = l_s_display_profile

EXCEPTIONS

profile_inconsistent = 1
internal_error = 2
no_data_available = 3
no_authority = 4

OTHERS = 5.

IF sy-subrc <> 0.

MESSAGE ID sy-msgid TYPE sy-msgty NUMBER sy-msgno

WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.

ENDIF.

l_s_display_profile-disvariant-report = sy-repid.

l_s_display_profile-disvariant-handle = 'LOG'.

CALL FUNCTION 'BAL_DSP_LOG_DISPLAY'

EXPORTING

i_s_display_profile = l_s_display_profile

EXCEPTIONS

OTHERS = 1.

IF sy-subrc <> 0.

MESSAGE ID sy-msgid TYPE sy-msgty NUMBER sy-msgno

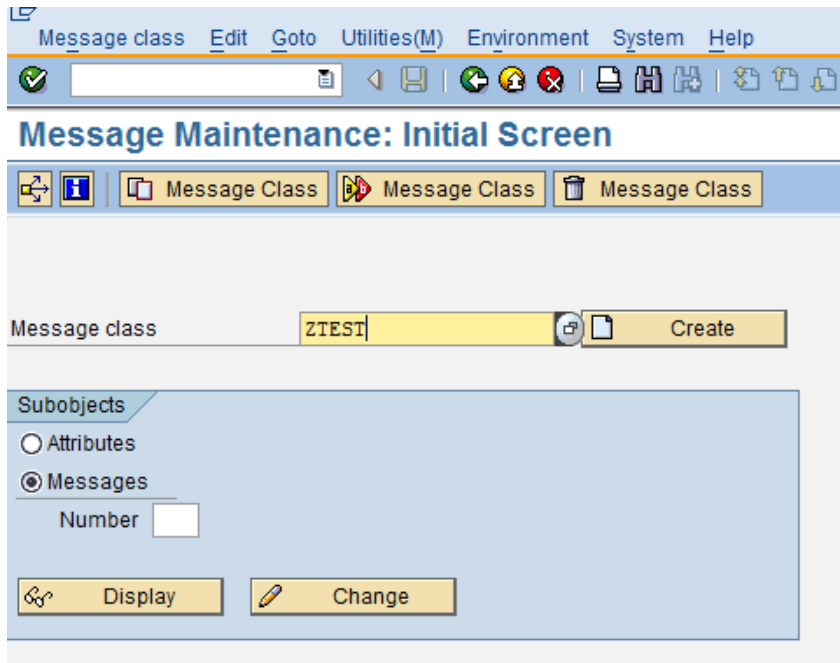
WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.

ENDIF.

How to create a Message class

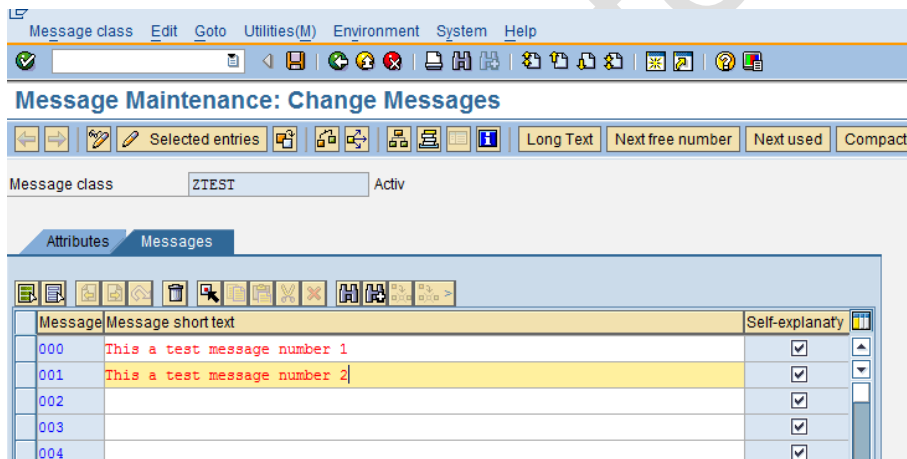
Message Class is a like a container which holds a number of different messages. Each message in the message class is identified with unique message number. So when you call a message in a [ABAP program](#), you need to specify the message class and message number.

Go to transaction SE91, enter the name of the message class and click on create button.



The screenshot shows the 'Message Maintenance: Initial Screen' in SAP. The 'Message class' field contains 'ZTEST'. Below it, the 'Subobjects' section has 'Messages' selected. The 'Number' field is empty. There are 'Display' and 'Change' buttons at the bottom of the subobjects section. A 'Create' button is next to the 'Message class' field.

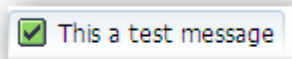
Maintain the required message texts with message numbers.



The screenshot shows the 'Message Maintenance: Change Messages' screen. The 'Message class' is 'ZTEST' and it is 'Activ'. The 'Messages' tab is selected. A table lists messages with their numbers and short texts. The first two messages are highlighted in yellow.

Message	Message short text	Self-explanatory
000	This a test message number 1	<input checked="" type="checkbox"/>
001	This a test message number 2	<input checked="" type="checkbox"/>
002		<input checked="" type="checkbox"/>
003		<input checked="" type="checkbox"/>
004		<input checked="" type="checkbox"/>

Messages can be issued as follows.
MESSAGE s000(ztest).



You can create message with parameters using C-language syntax.

Message Maintenance: Change Messages

Message class: Activ.

Attributes Messages

Message	Message Short Text	Self-Explanatory
000	Search results: &1 entry(ies) was found.	<input checked="" type="checkbox"/>
001	This a test message number 2	<input checked="" type="checkbox"/>
002	This is a test message with parametres &1 and &2	<input checked="" type="checkbox"/>
003		<input checked="" type="checkbox"/>

Symbol "&" determines the spot in witch string argument will be inserted.

Messages with parameters can be used as follows:

`MESSAGE s000(zlx_mc) WITH `1`.`

Typ	Message text
<input checked="" type="checkbox"/>	Search results: 1 entry(ies) was found.

How to create a GUI-status

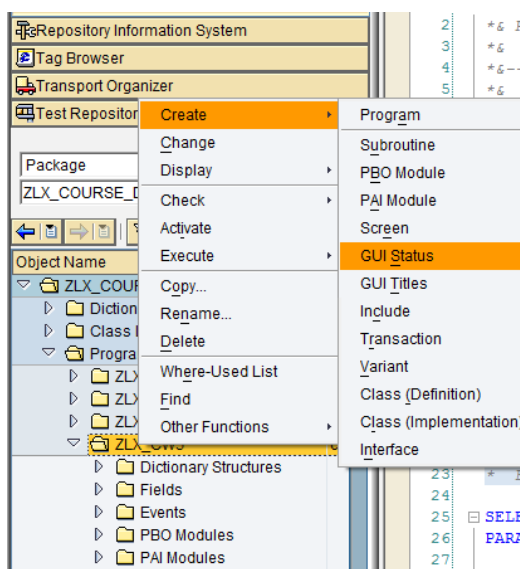
The function of a GUI status is to provide the user with a range of functions on a screen.

Each function has an associated function code of up to 20 characters, and when the user chooses a function, the PAI event is triggered.

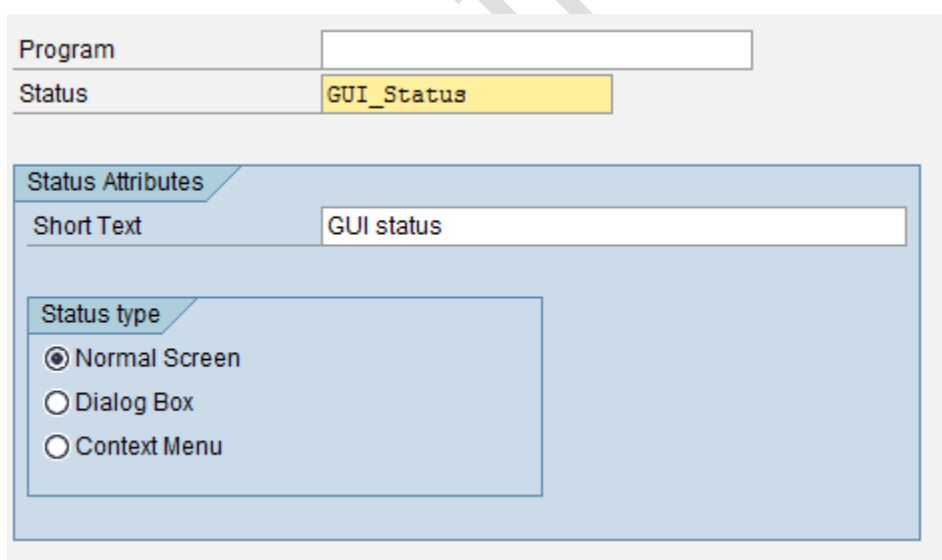
In each PAI event, the function code, as long as it is not empty, is placed in the system field SYST-UCOMM (SY-UCOMM) and assigned to the OK_CODE field. Empty function codes are placed in neither the SY-UCOMM field nor the OK_CODE field. Before you can work with the OK_CODE field, you must assign a name to it in the Screen Painter.

All function codes in an ABAP program, apart from those only assigned to pushbuttons on screens, are defined and administered in the Menu Painter.

Go to transaction SE80, select your program, right click on it choose Create->GUI Status

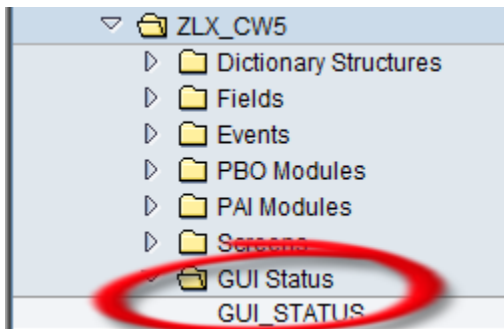


Enter status name and Short text, click Continue

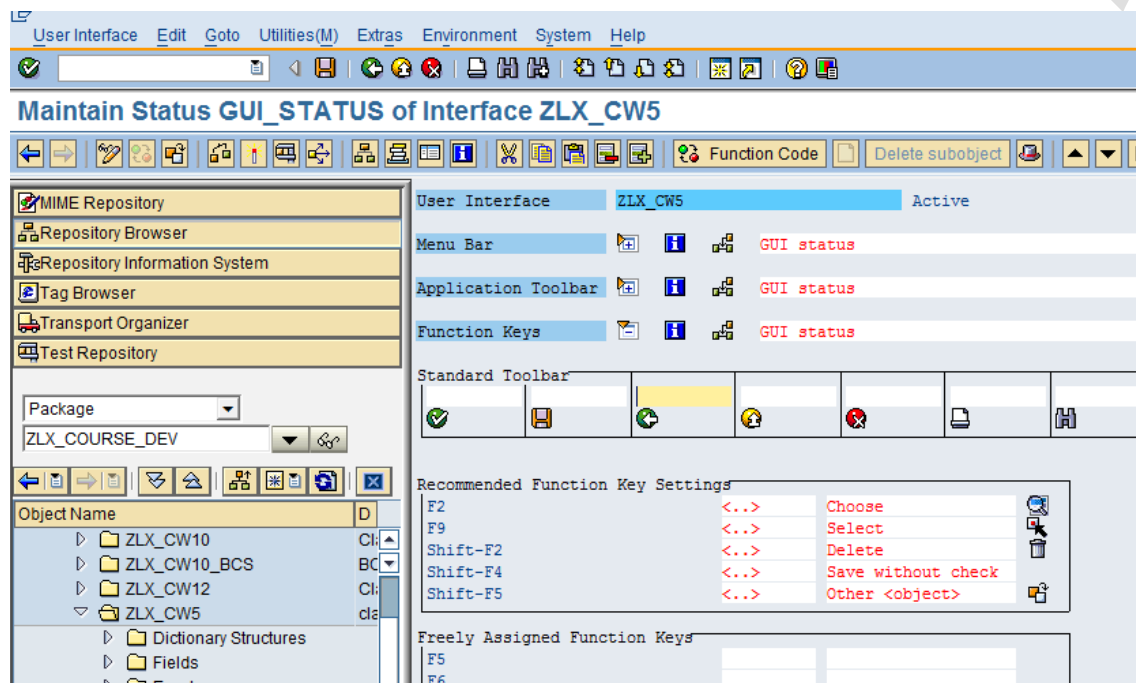


Save and Activate it.

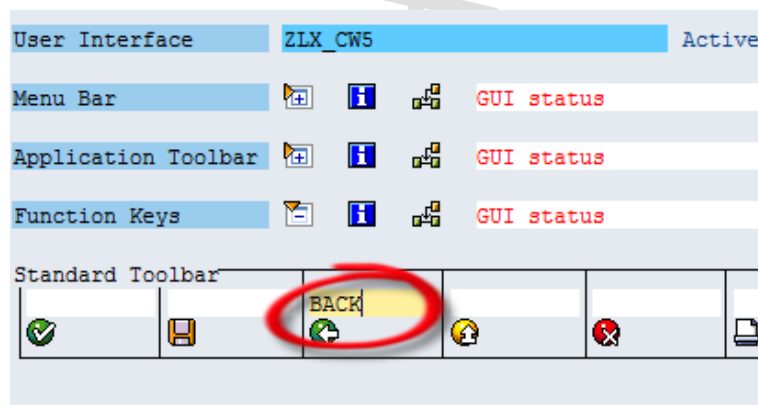
You can see GUI-status in your program in SE80 transaction.



Double click on your GUI-status and expand *Function Keys*



Enter name for button, save and activate it.



In the PBO module you need set pf-status:

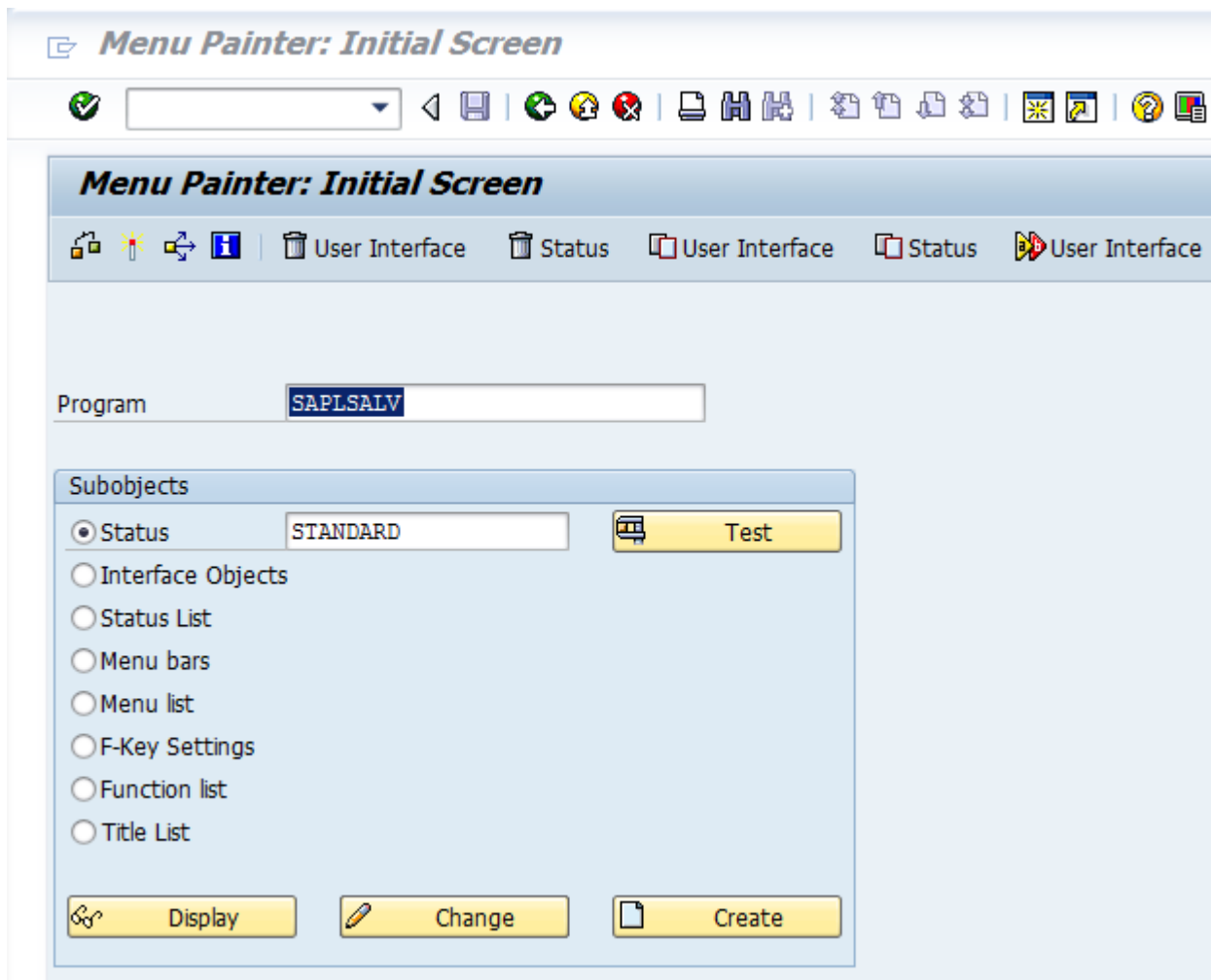
```
SET PF-STATUS 'GUI_STATUS'.
```

In the PAI module you need handle the event of your button:

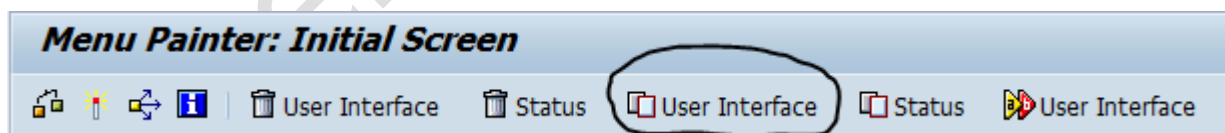
```
CASE sy-ucomm.  
  WHEN 'BACK'.  
    LEAVE TO SCREEN 0.
```

If you want to use and modify prepared GUI status that's already implemented in other programs, you need to do following:

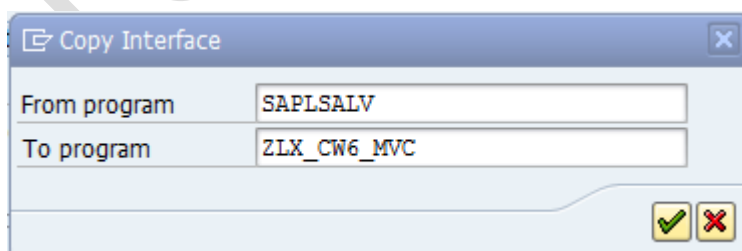
- 1) Go to transaction SE41.
- 2) Enter program name and GUI status name witch you want to copy in your Z* program:
(Example GUI status for ALV Grid)



- 3) Press User Interface button witch signed as "Copy user interface" (Ctrl+F5 combination)



- 4) Enter name of your Z* program in popup window:



After that GUI status appears in corresponding program folder:

<ul style="list-style-type: none"> <ul style="list-style-type: none"> Dictionary Structure Fields Events Screens GUI Status <ul style="list-style-type: none"> STANDARD STDPOPUP 	<p>Classwork 6 - MVC design pattern usage demon</p> <p>Standard for General List Output</p> <p>Standard for General List Output in Dialog Box</p>
---	---

5) Double click on STANDARD and you will see prepared content of GUI Status.

Display Status STANDARD, Interface ZLX_CW6_MVC

Connectivity Browser | MIME Repository | Repository Browser | Repository Information System | Tag Browser | Transport Organizer | Test Repository

Package: ZLX_COURSE_MAIN

Object Name | Description

Programs	
ZLX_CW10	ClassWork 10 - Smartforms and PDF-forms
ZLX_CW10_BCS	BCS
ZLX_CW12	ClassWork 12 - Web Services
ZLX_CW5	Classwork 5 - Sel. scr.(parametres), Custom scr.,
ZLX_CW5_SO	Classwork 5 - Selection screen (select-opt.), BA
Dictionary Structure	
Fields	
Events	
ZLX_CW6	Class work 6
ZLX_CW6_MVC	Classwork 6 - MVC design pattern usage demon
Dictionary Structure	
Fields	
Events	
Screens	
GUI Status	
STANDARD	Standard for General List Output
STDPOPUP	Standard for General List Output in Dialog Box

User Interface: ZLX_CW6_MVC Inactive

Menu Bar: Standard User Interface STANDARD

Application Toolbar: Standard Maximum Interaction

Items 1 - 7	ETA	EB9	ALL	SAL	OUP
Items 8 - 14	ODN	ILT	UMC	SUM	XPA
Items 15 - 21	XXL	AW	PC	SL	ABC
Items 22 - 28	OLO	OAD	AVE	LFO	INFO Select
Items 29 - 35	CRB	CRL	CRR	CRE	

Function Keys: Standard Maximum Interaction

Standard Toolbar: F03, F15, F12, RNT, SC, SC+, P-, P-

Reserved Function Keys: F4, Shift-F10

Recommended Function Key Settings

6) Modify buttons according to your program logic, save and activate it.