How to transfer custom fields from SRM to ECC

Applies to:

SRM5 <-> SAP ECC6.

Summary

In the shopping cart of your SRM system some custom fields in the position of the requisition are required.

The field-content entered during purchasing shall be transferred to the according customer fields in the purchase requisition or purchase order in the backend system.

In this example two customer fields are created in both SRM and backend and are transferred to the Purchase requisition respectively purchase order.

Author(s): Andreas Milbredt

Company: Acando GmbH, Hamburg http://www.acando.de

Created on: 20 May 2009

Author Bio

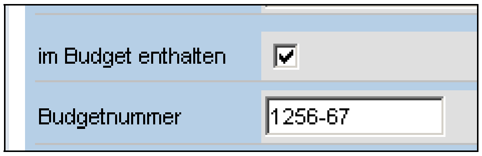
Working as a SAP consultant since 1996 with some exkursions to IBM's WebSphere world in former times, since 3 years now with Acando Hamburg. Main topics at the moment are Netweaver Portal, WebDynpro for ABAP and SRM.

Table of Contents

Requirement

In the shopping cart of your SRM system you need some custom fields in the position of the requisition.

SRM:

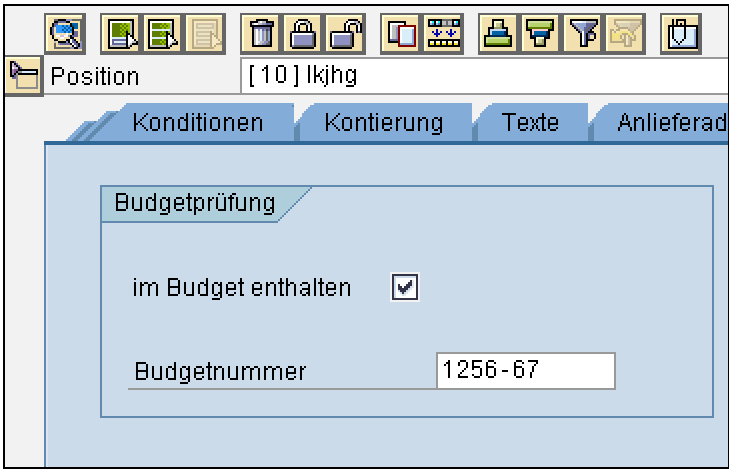


im Budget enthalten ☑

Budgetnummer 1256-67

The field-content entered during purchasing shall be transferred to the according customer fields in the purchase requisition or purchase order in the backend system.

Purchase order:



Position [10] lkjhg

Budgetprüfung

im Budget enthalten ☑

Budgetnummer 1256-67

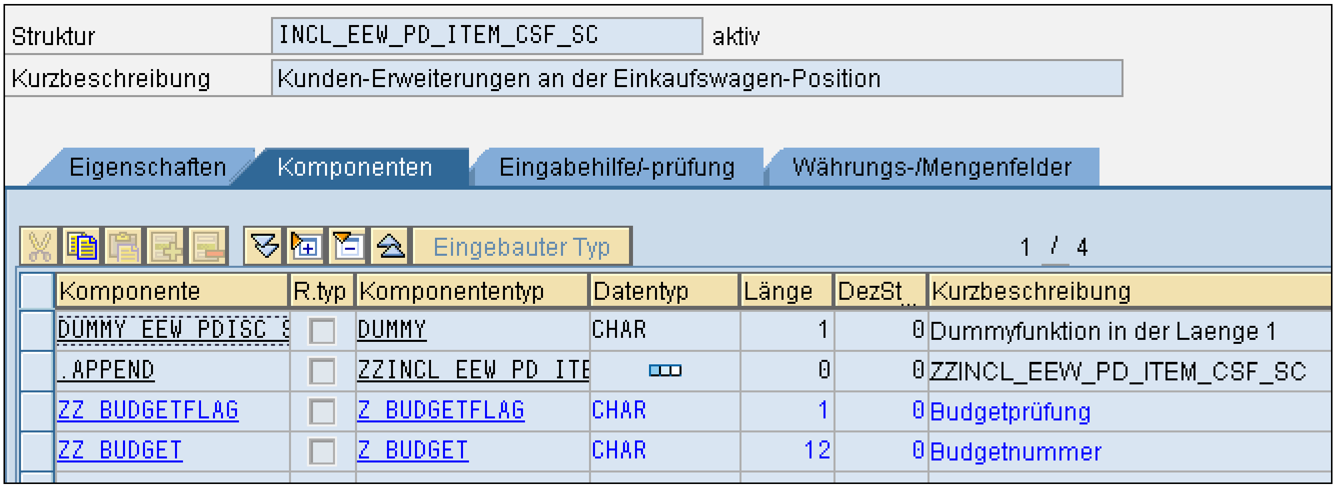
In this example 2 customer fields are created.

Solution

Create a customer field in the shopping cart

Look at Note 672960 for SRM5 (for SRM < 4.0 SAPNOTE 458591) how to create a customer field

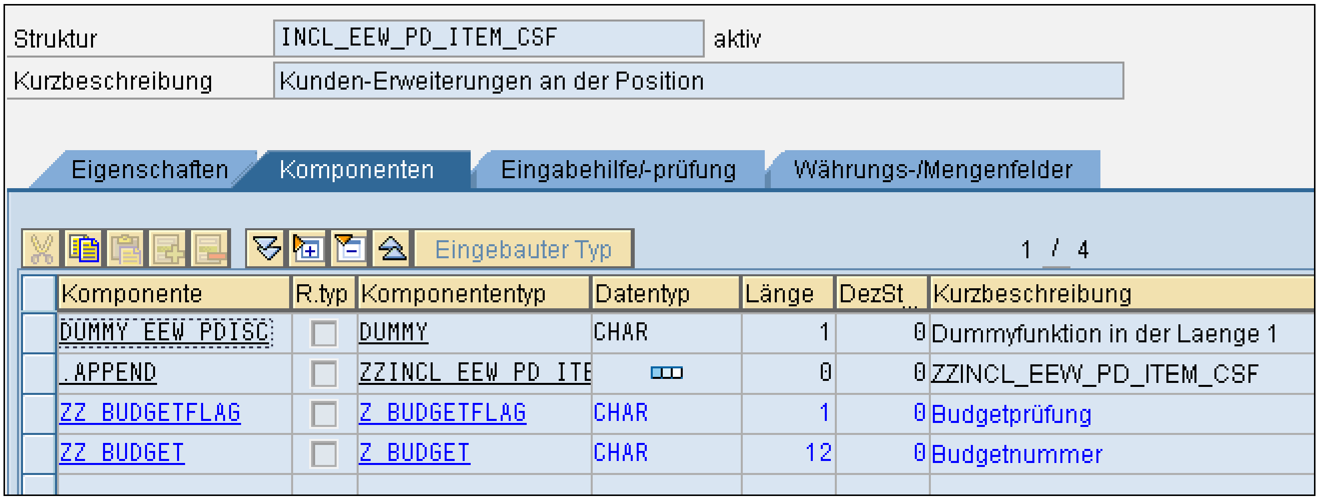
Transaction SE11: Create an append structure for “INCL_EEW_PD_ITEM_CSF_SC”



Struktur INCL_EEW_PD_ITEM_CSF_SC aktiv
Kurzbeschreibung Kunden-Erweiterungen an der Einkaufswagen-Position

Komponente	R.typ	Komponententyp	Datentyp	Länge	DezSt...	Kurzbeschreibung
DUMMY_EEW_PDISC_S	☐	DUMMY	CHAR	1	0	Dummyfunktion in der Laenge 1
.APPEND	☐	ZZINCL_EEW_PD_ITE	▭	0	0	ZZINCL_EEW_PD_ITEM_CSF_SC
ZZ_BUDGETFLAG	☐	Z_BUDGETFLAG	CHAR	1	0	Budgetprüfung
ZZ_BUDGET	☐	Z_BUDGET	CHAR	12	0	Budgetnummer

Also create an append structure for „INCL_EEW_PD_ITEM_CSF“ (When this append does not exist, the values entered in the shopping cart are not saved)



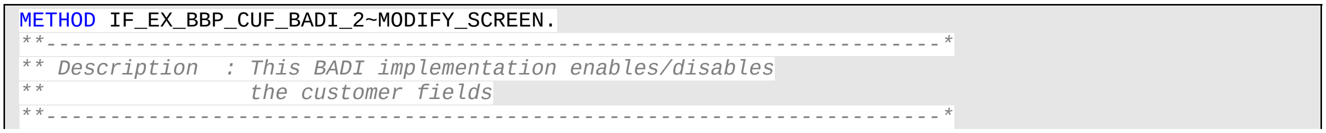
Struktur INCL_EEW_PD_ITEM_CSF aktiv
Kurzbeschreibung Kunden-Erweiterungen an der Position

Komponente	R.typ	Komponententyp	Datentyp	Länge	DezSt...	Kurzbeschreibung
DUMMY_EEW_PDISC	☐	DUMMY	CHAR	1	0	Dummyfunktion in der Laenge 1
.APPEND	☐	ZZINCL_EEW_PD_ITE	▭	0	0	ZZINCL_EEW_PD_ITEM_CSF
ZZ_BUDGETFLAG	☐	Z_BUDGETFLAG	CHAR	1	0	Budgetprüfung
ZZ_BUDGET	☐	Z_BUDGET	CHAR	12	0	Budgetnummer

For the customer field ZZ_BUDGETFLAG here a component type Z_BUDGETFLAG has been created, that points to the domain „XFELD“, this ensures that the field appears on the screen as a checkbox.

Code sample, optional – the fields are only shown during approval of the shopping cart

In SE19 implement the BAdi BBP_CUF_BADI_2, method MODIFY_SCREEN



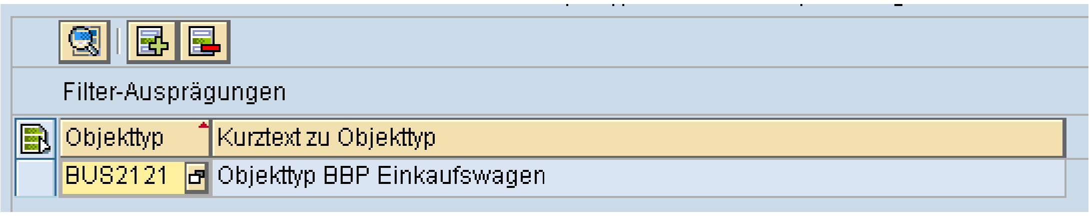
```
METHOD IF_EX_BBP_CUF_BADI_2~MODIFY_SCREEN.
**---------------------------------------------------------------------------*
** Description  : This BADI implementation enables/disables
**                the customer fields
**---------------------------------------------------------------------------*
```

```
  DATA: WA_FIELDS         TYPE BBPS_CUF_DISPLAY.
  DATA: LT_ITEM           TYPE TABLE OF BBP_PDS_SC_ITEM_D.
  DATA: WA_ITEM           TYPE BBP_PDS_SC_ITEM_D.
  DATA: LT_MESSAGES       TYPE TABLE OF BBP_PDS_MESSAGES.
  DATA: WA_MESSAGES       LIKE LINE OF LT_MESSAGES.
  DATA: LT_STATUS TYPE    TABLE OF BBP_PDS_STATUS.
  DATA: WA_STATUS LIKE    LINE OF  LT_STATUS.
  DATA: LV_STATUS_APPROVE TYPE C.

* The Customer fields are enabled by default and their attributes are passed
* in the parameter ET_fields. To disable the Customer fields set the
* XINPUT and XDISPLAY value to space.

* Nur bei Genehmigungsschritt soll das Feld angezeigt werden
  CALL FUNCTION 'BBP_PD_SC_GETDETAIL'
    EXPORTING
      I_GUID          = IV_DOC_GUID
      I_WITH_ITEMDATA = 'X'
    TABLES
      E_ITEM          = LT_ITEM
      E_MESSAGES      = LT_MESSAGES
      E_STATUS        = LT_STATUS.
  .
* Feld soll nur bei Status "In Genehmigung" angezeigt werden (lt.Tabelle TJ02)
  LOOP AT LT_STATUS INTO WA_STATUS.
    IF WA_STATUS-STAT EQ 'I1015' AND WA_STATUS-INACT IS INITIAL.
      LV_STATUS_APPROVE = 'X'.
    ENDIF.
  ENDLOOP.

* Feld sichtbar/unsichtbar
  LOOP AT ET_FIELDS INTO WA_FIELDS.
* Budgetflag
    IF WA_FIELDS-FIELDNAME = 'ZZ_BUDGETFLAG'.
      IF  LV_STATUS_APPROVE IS INITIAL.
        WA_FIELDS-XDISPLAY = ''.
        MODIFY ET_FIELDS FROM WA_FIELDS.
* Feld eingabebereit
      ELSEIF LV_STATUS_APPROVE EQ 'X'.
        WA_FIELDS-XINPUT = 'X'.
        MODIFY ET_FIELDS FROM WA_FIELDS.
      ENDIF.
    ENDIF.
* Budgetnummer
    IF WA_FIELDS-FIELDNAME = 'ZZ_BUDGET'.
      IF  LV_STATUS_APPROVE IS INITIAL.
        WA_FIELDS-XDISPLAY = ''.
        MODIFY ET_FIELDS FROM WA_FIELDS.
* Feld eingabebereit
      ELSEIF LV_STATUS_APPROVE EQ 'X'.
        WA_FIELDS-XINPUT = 'X'.
        MODIFY ET_FIELDS FROM WA_FIELDS.
      ENDIF.
    ENDIF.
  ENDLOOP.

ENDMETHOD.
```

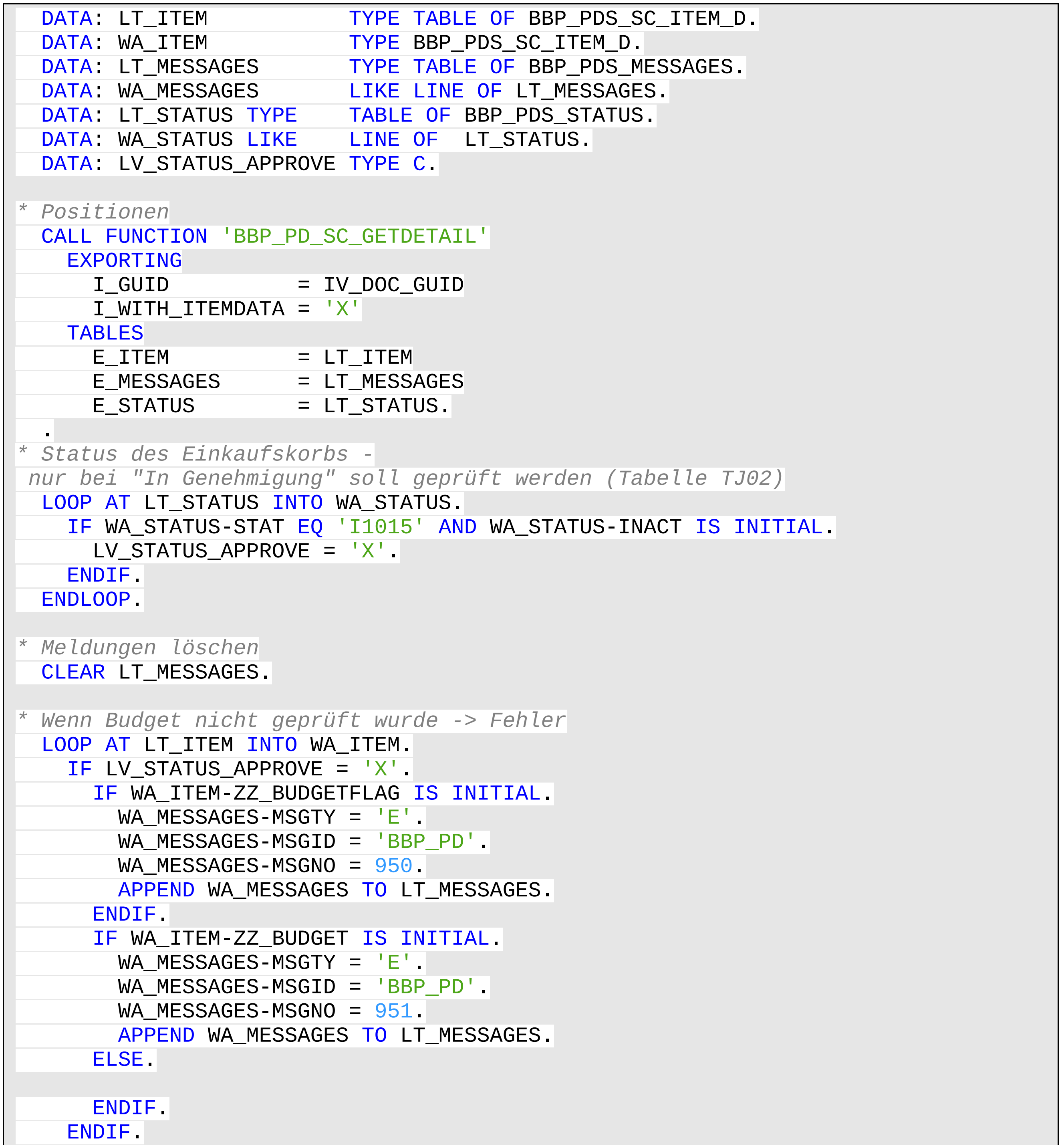
Code sample, optional – check if fields have been edited

In SE19 implement the BAdi BBP_ITEM_CHECK_BADI, method BBP_ITEM_CHECK

Set the filter


Filter-Ausprägungen

Objekttyp	Kurztext zu Objekttyp
BUS2121	Objekttyp BBP Einkaufswagen


```
  DATA: LT_ITEM             TYPE TABLE OF BBP_PDS_SC_ITEM_D.
  DATA: WA_ITEM             TYPE BBP_PDS_SC_ITEM_D.
  DATA: LT_MESSAGES         TYPE TABLE OF BBP_PDS_MESSAGES.
  DATA: WA_MESSAGES         LIKE LINE OF LT_MESSAGES.
  DATA: LT_STATUS TYPE      TABLE OF BBP_PDS_STATUS.
  DATA: WA_STATUS LIKE      LINE OF  LT_STATUS.
  DATA: LV_STATUS_APPROVE TYPE C.

* Positionen
  CALL FUNCTION 'BBP_PD_SC_GETDETAIL'
    EXPORTING
      I_GUID            = IV_DOC_GUID
      I_WITH_ITEMDATA = 'X'
    TABLES
      E_ITEM            = LT_ITEM
      E_MESSAGES        = LT_MESSAGES
      E_STATUS          = LT_STATUS.
  .
* Status des Einkaufskorbs -
 nur bei "In Genehmigung" soll geprüft werden (Tabelle TJ02)
  LOOP AT LT_STATUS INTO WA_STATUS.
    IF WA_STATUS-STAT EQ 'I1015' AND WA_STATUS-INACT IS INITIAL.
      LV_STATUS_APPROVE = 'X'.
    ENDIF.
  ENDLOOP.

* Meldungen löschen
  CLEAR LT_MESSAGES.

* Wenn Budget nicht geprüft wurde -> Fehler
  LOOP AT LT_ITEM INTO WA_ITEM.
    IF LV_STATUS_APPROVE = 'X'.
      IF WA_ITEM-ZZ_BUDGETFLAG IS INITIAL.
        WA_MESSAGES-MSGTY = 'E'.
        WA_MESSAGES-MSGID = 'BBP_PD'.
        WA_MESSAGES-MSGNO = 950.
        APPEND WA_MESSAGES TO LT_MESSAGES.
      ENDIF.
      IF WA_ITEM-ZZ_BUDGET IS INITIAL.
        WA_MESSAGES-MSGTY = 'E'.
        WA_MESSAGES-MSGID = 'BBP_PD'.
        WA_MESSAGES-MSGNO = 951.
        APPEND WA_MESSAGES TO LT_MESSAGES.
      ELSE.

      ENDIF.
    ENDIF.
  ENDLOOP.

* Meldungen ins Log, falls vorhanden
  CALL FUNCTION 'BBP_PDH_PO_ADD_TO_MSG_LOG'
    TABLES
      IT_MESSAGE = LT_MESSAGES.
```

Code sample, optional – set shopping cart to edit mode during approval

Pls see also this SDN-thread: https://www.sdn.sap.com/irj/scn/thread?forumID=346&threadID=1317783

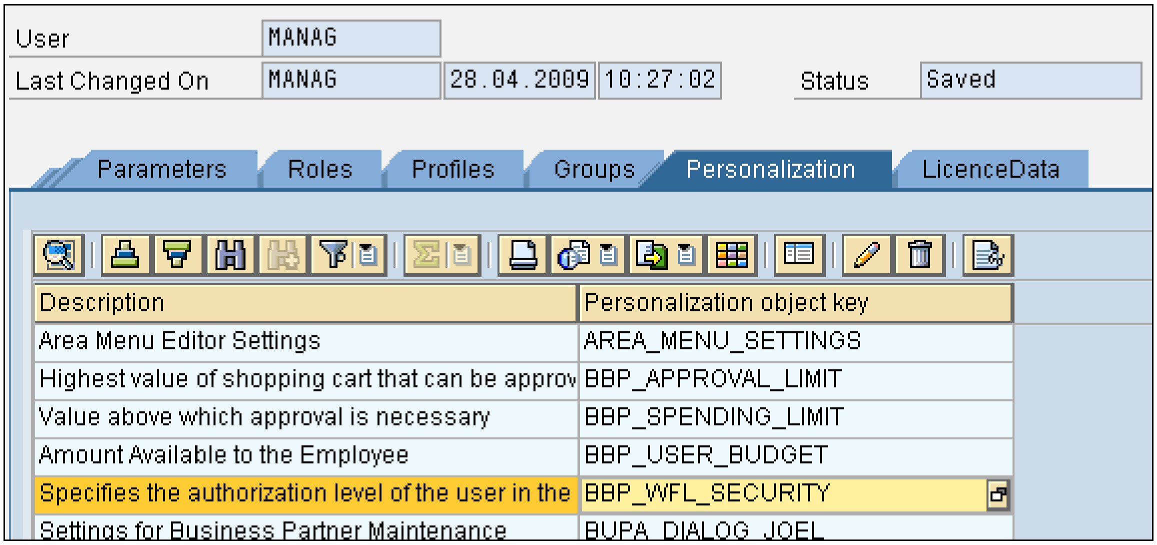
In SE19 replace the standard implementation „BBP_WFL_SECUR_BADI_S“ of BAdi „BBP_WFL_SECUR_BADI“, method „SET_SECURITY_LEVEL“by an own.

Here the only difference between the standard implementation and the z-implementation the if-clause „if scenario is initial“ is commented out.

```
method IF_EX_BBP_WFL_SECUR_BADI~SET_SECURITY_LEVEL.

  if object_type eq 'BUS2121'.
*** This BADI is called in two main scenarios:
*** 1) To determine if the business object could be changed. The
***    variable scenario is filled in this case.
*** 2) To find out if the workflow should be restarted ( in
***    case of security level "medium" this depends also from
***    starting conditions.
*** The variable scenario is initial only in case 2)
*    if scenario is initial .
      call function 'BBP_PDH_WFL_SECLEVEL_DB_GET'
        exporting
          iv_user      = actual_user
        importing
          ev_sec_level = new_sec_level.

      if new_sec_level is initial.
        new_sec_level = 4.
      endif.
*    endif.
  endif.
```

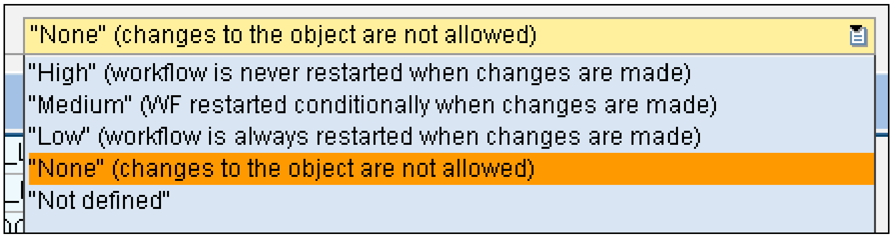
Using this implementation the personalization of the User (in SU01) does exactly what it promises.

Personalization object key: BBP_WFL_SECURITY



User	MANAG			
Last Changed On	MANAG	28.04.2009	10:27:02	Status: Saved

Description	Personalization object key
Area Menu Editor Settings	AREA_MENU_SETTINGS
Highest value of shopping cart that can be approv	BBP_APPROVAL_LIMIT
Value above which approval is necessary	BBP_SPENDING_LIMIT
Amount Available to the Employee	BBP_USER_BUDGET
Specifies the authorization level of the user in the	BBP_WFL_SECURITY
Settings for Business Partner Maintenance	BUPA_DIALOG_JOEL



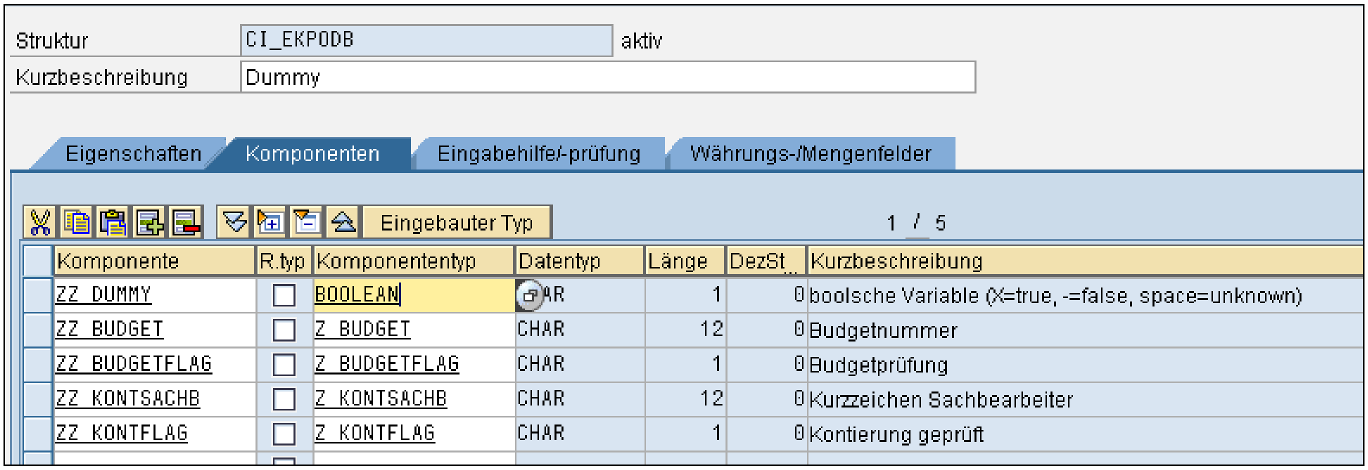
"None" (changes to the object are not allowed)

- "High" (workflow is never restarted when changes are made)
- "Medium" (WF restarted conditionally when changes are made)
- "Low" (workflow is always restarted when changes are made)
- "None" (changes to the object are not allowed)
- "Not defined"

Create Customer fields fort he purchase order in the Backend

Read the documentation of Enhancement MM06E005 (Transaction SMOD)

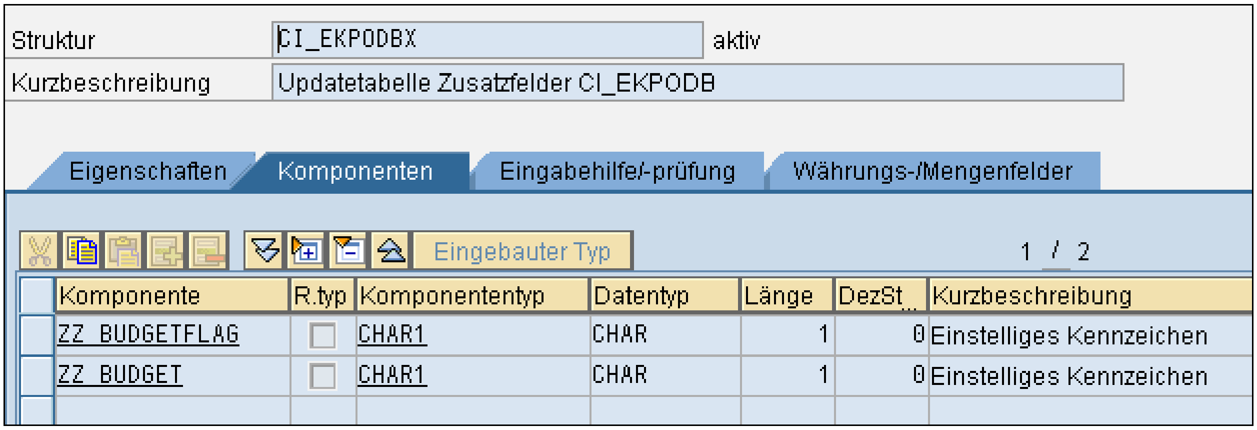
Create both fields (ZZ_BUDGETFLAG, ZZ_BUDGET) through creation of structure CI_EKPODB in SE11



Struktur	CI_EKPODB	aktiv
Kurzbeschreibung	Dummy	

Komponente	R.typ	Komponententyp	Datentyp	Länge	DezSt...	Kurzbeschreibung
ZZ_DUMMY	□	BOOLEAN	AR	1	0	boolsche Variable (X=true, -=false, space=unknown)
ZZ_BUDGET	□	Z_BUDGET	CHAR	12	0	Budgetnummer
ZZ_BUDGETFLAG	□	Z_BUDGETFLAG	CHAR	1	0	Budgetprüfung
ZZ_KONTSACHB	□	Z_KONTSACHB	CHAR	12	0	Kurzzeichen Sachbearbeiter
ZZ_KONTFLAG	□	Z_KONTFLAG	CHAR	1	0	Kontierung geprüft

Also create the structure „CI_EKPODBX“ in SE11. This structure is needed lateron for transfer of the fields from SRM to the backend using „BAPI_PO_CREATE1“.

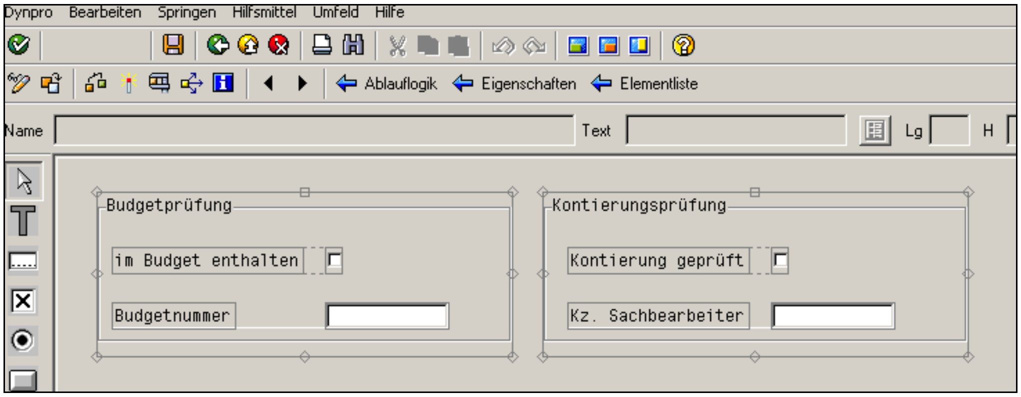


Struktur CI_EKPODBX aktiv

Kurzbeschreibung Updatetabelle Zusatzfelder CI_EKPODB

Komponente	R.typ	Komponententyp	Datentyp	Länge	DezSt...	Kurzbeschreibung
ZZ_BUDGETFLAG	□	CHAR1	CHAR	1	0	Einstelliges Kennzeichen
ZZ_BUDGET	□	CHAR1	CHAR	1	0	Einstelliges Kennzeichen

Create Dynpro SAPLXM06 and activate it



Budgetprüfung

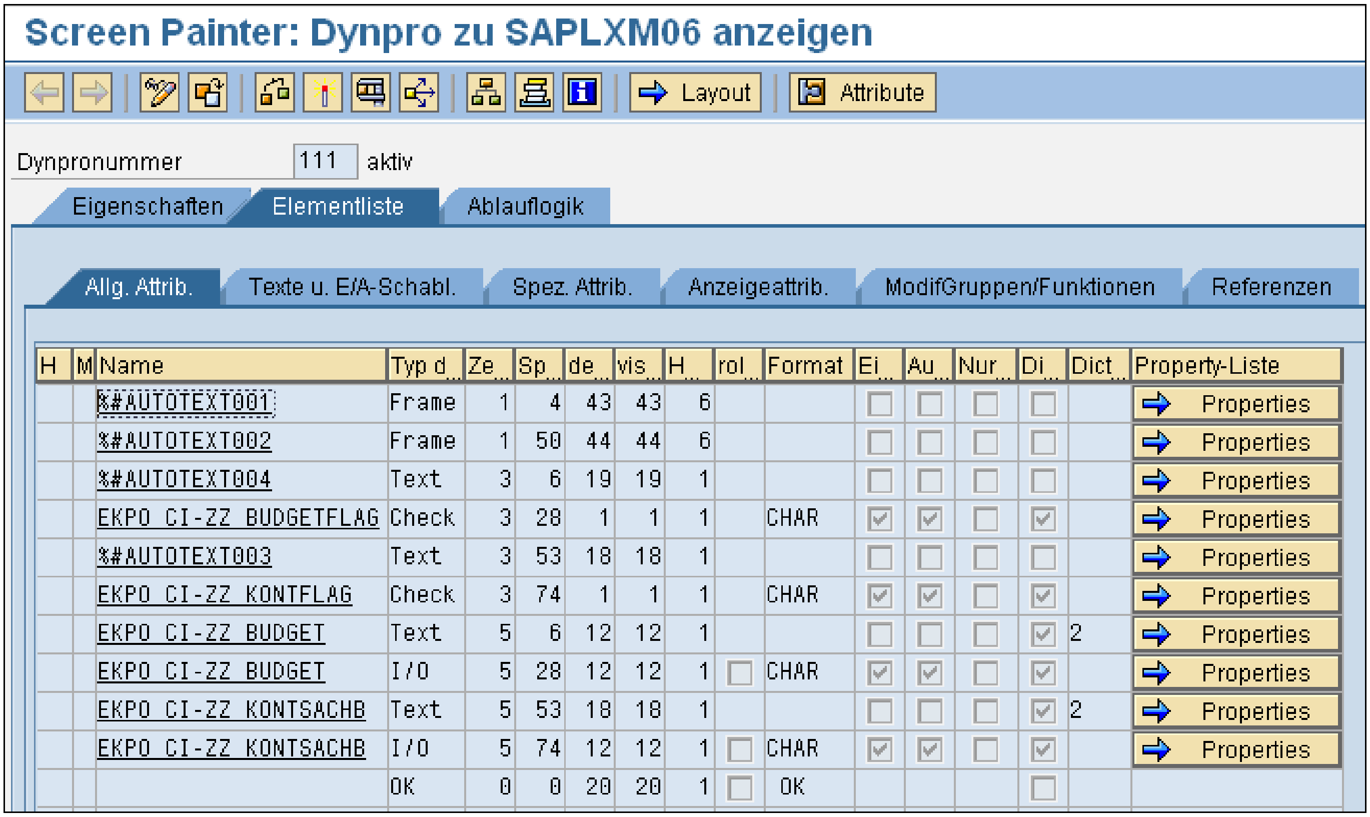
im Budget enthalten □

Budgetnummer

Kontierungsprüfung

Kontierung geprüft □

Kz. Sachbearbeiter

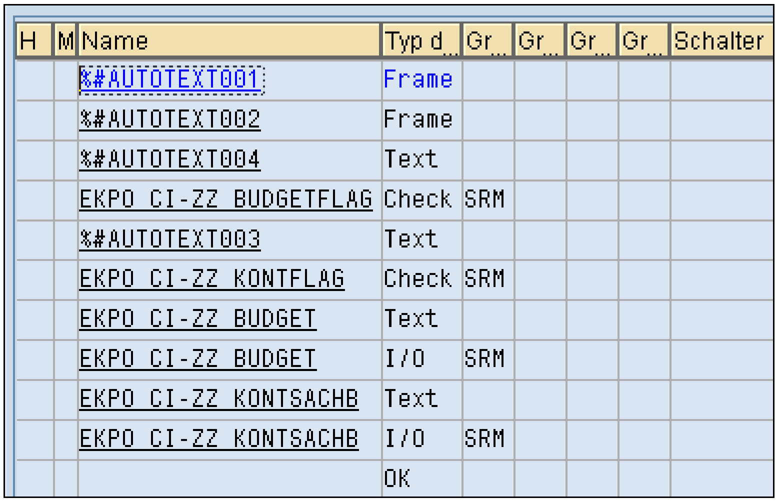


Screen Painter: Dynpro zu SAPLXM06 anzeigen

Dynpronummer 111 aktiv

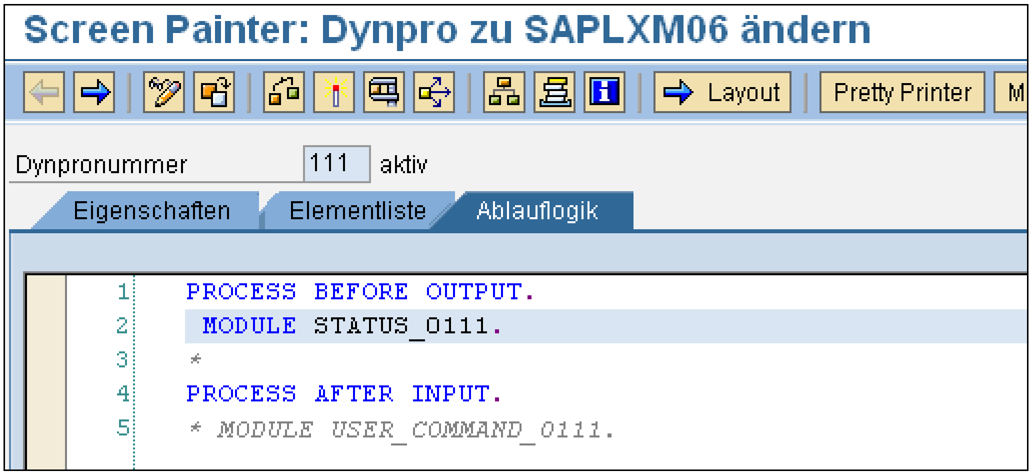
H	M	Name	Typ d...	Ze...	Sp...	de...	vis...	H...	rol...	Format	Ei...	Au...	Nur...	Di...	Dict...	Property-Liste
		%#AUTOTEXT001	Frame	1	4	43	43	6			□	□	□	□		Properties
		%#AUTOTEXT002	Frame	1	50	44	44	6			□	□	□	□		Properties
		%#AUTOTEXT004	Text	3	6	19	19	1			□	□	□	□		Properties
		EKPO CI-ZZ BUDGETFLAG	Check	3	28	1	1	1		CHAR	☑	☑	□	☑		Properties
		%#AUTOTEXT003	Text	3	53	18	18	1			□	□	□	□		Properties
		EKPO CI-ZZ KONTFLAG	Check	3	74	1	1	1		CHAR	☑	☑	□	☑		Properties
		EKPO CI-ZZ BUDGET	Text	5	6	12	12	1			□	□	□	☑	2	Properties
		EKPO CI-ZZ BUDGET	I/O	5	28	12	12	1	□	CHAR	☑	☑	□	☑		Properties
		EKPO CI-ZZ KONTSACHB	Text	5	53	18	18	1			□	□	□	☑	2	Properties
		EKPO CI-ZZ KONTSACHB	I/O	5	74	12	12	1	□	CHAR	☑	☑	□	☑		Properties
			OK	0	0	20	20	1	□	OK				□		

Modification group SRM



H	M	Name	Typ d...	Gr...	Gr...	Gr...	Gr...	Schalter
		%#AUTOTEXT001	Frame					
		%#AUTOTEXT002	Frame					
		%#AUTOTEXT004	Text					
		EKPO CI-ZZ BUDGETFLAG	Check	SRM				
		%#AUTOTEXT003	Text					
		EKPO CI-ZZ KONTFLAG	Check	SRM				
		EKPO CI-ZZ BUDGET	Text					
		EKPO CI-ZZ BUDGET	I/O	SRM				
		EKPO CI-ZZ KONTSACHB	Text					
		EKPO CI-ZZ KONTSACHB	I/O	SRM				
			OK					

create PBO Modul



```
PROCESS BEFORE OUTPUT.
 MODULE STATUS_0111.
*
PROCESS AFTER INPUT.
* MODULE USER_COMMAND_0111.
```

(In the Topinclude of the exits belonging to SAPLMX06 a field „gv_trtyp“ is defined.The field holds the activity mode (display, change)

```
  gv_trtyp = i_trtyp.
```

So the value can be used in the PBO module to determine the mode and to set the screen-input accordingly

```
module STATUS_0111 output.

* Feldsteuerung
  LOOP AT SCREEN.

* when display-mode
    IF screen-group1 = 'SRM'.
      IF gv_trtyp = 'A'.
        screen-input = 0.
      ELSE.
        screen-input = 1.
      ENDIF.
      MODIFY SCREEN.
    ENDIF.

  ENDLOOP.

*   SET PF-STATUS 'xxxxxxxx'.
*   SET TITLEBAR 'xxx'.

endmodule.                 " STATUS_0111   OUTPUT
```

In SAP-Enhancement MM06E005 implement the exit „EXIT_SAPMM06E_016“, Include „ZXM06U41“ (PBO)

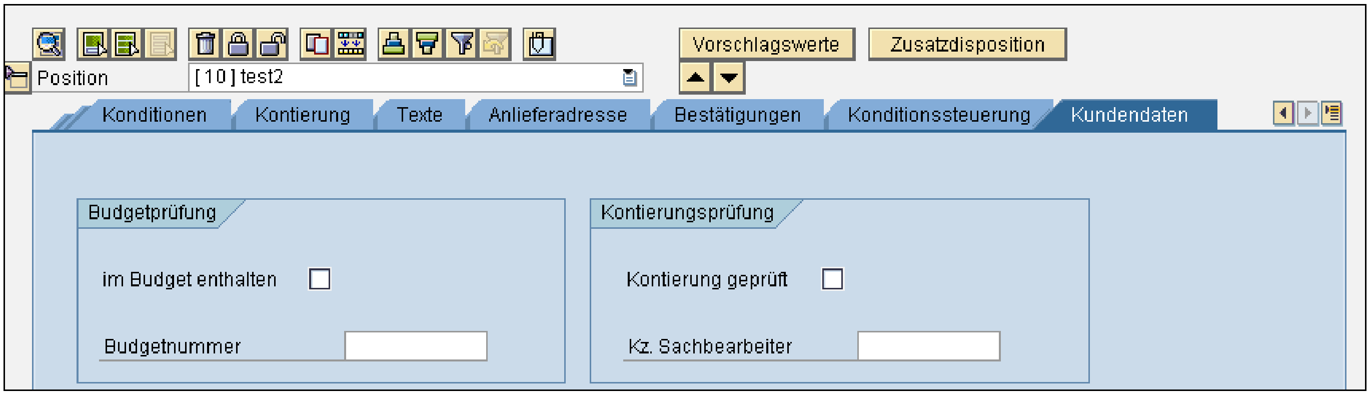
```
*&---------------------------------------------------------------------*
*&  Include           ZXM06U41
*&---------------------------------------------------------------------*
```

```
* DB -> Screen
move-corresponding i_ekpo to ekpo_ci.
```

In SAP-Enhancement MM06E005 implement the exit „EXIT_SAPMM06E_018“, Include„ZXM06U40“ (PAI)

```
*&---------------------------------------------------------------------*
*&  Include           ZXM06U40
*&---------------------------------------------------------------------*


if e_ci_ekpo <> ekpo_ci.
  e_ci_update = 'X'.
  move-corresponding ekpo_ci to e_ci_ekpo.
endif.
```

Looks like this in the PO:



Position [10] test2

Budgetprüfung

im Budget enthalten □

Budgetnummer

Kontierungsprüfung

Kontierung geprüft □

Kz. Sachbearbeiter

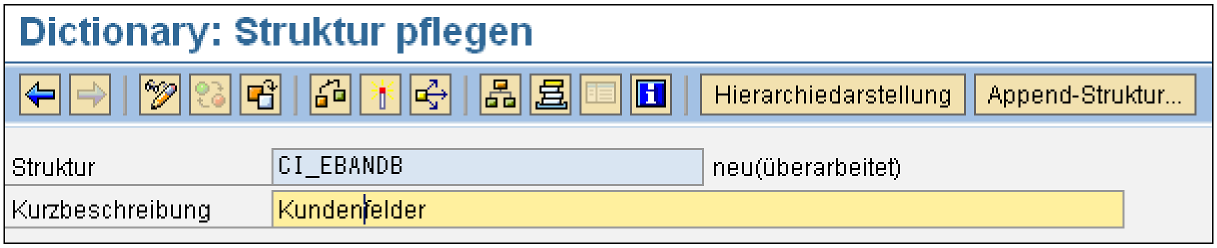
activate the enhancement

Create customer fields in the purchase requisition in the backend

Read the documentation of Enhancement MEREQ001 (Transaction SMOD)

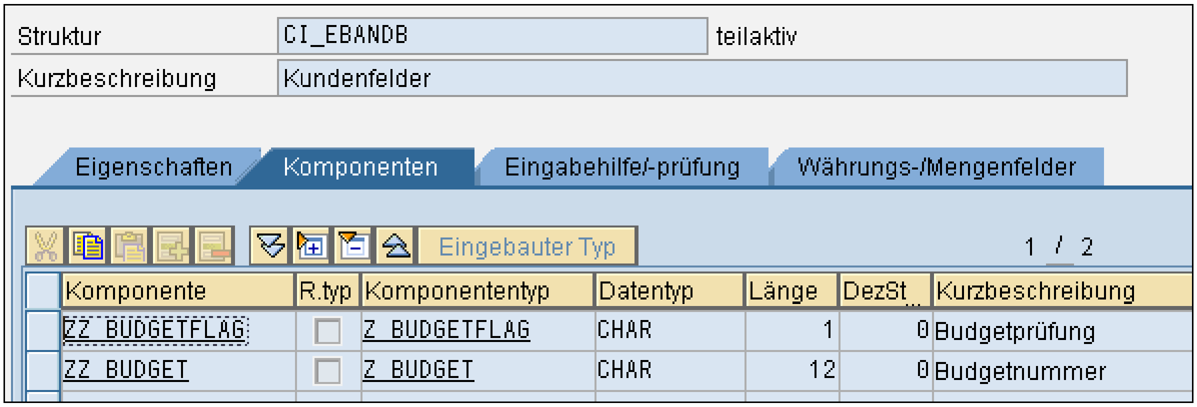
In SE11 display tabke EBAN, create structure CI_EBANDB

Here no structure CI_EBANDBX is needed



Dictionary: Struktur pflegen

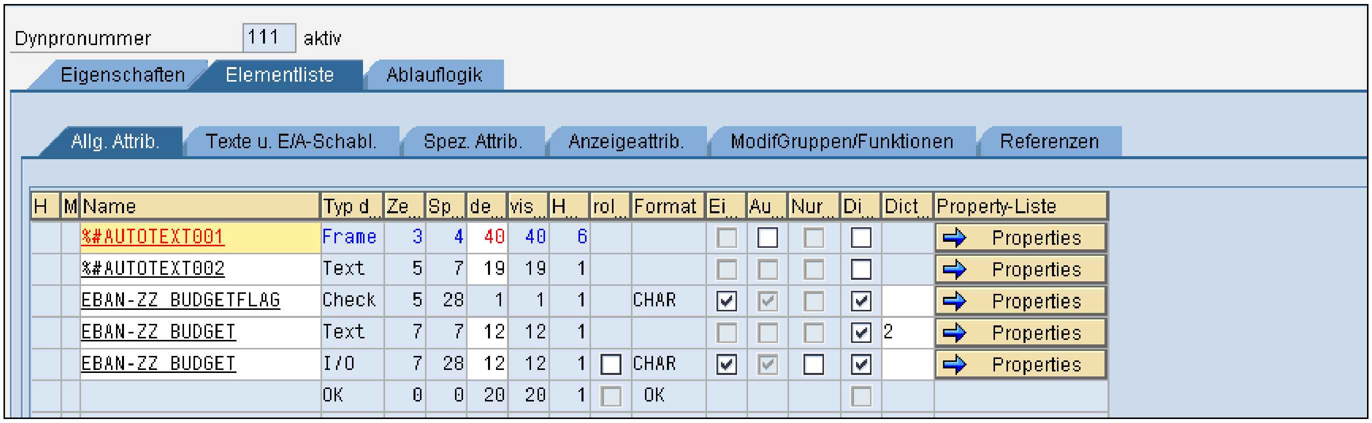
Struktur	CI_EBANDB	neu(überarbeitet)
Kurzbeschreibung	Kundenfelder	



Struktur	CI_EBANDB	teilaktiv
Kurzbeschreibung	Kundenfelder	

Komponente	R.typ	Komponententyp	Datentyp	Länge	DezSt...	Kurzbeschreibung
ZZ_BUDGETFLAG	□	Z_BUDGETFLAG	CHAR	1	0	Budgetprüfung
ZZ_BUDGET	□	Z_BUDGET	CHAR	12	0	Budgetnummer

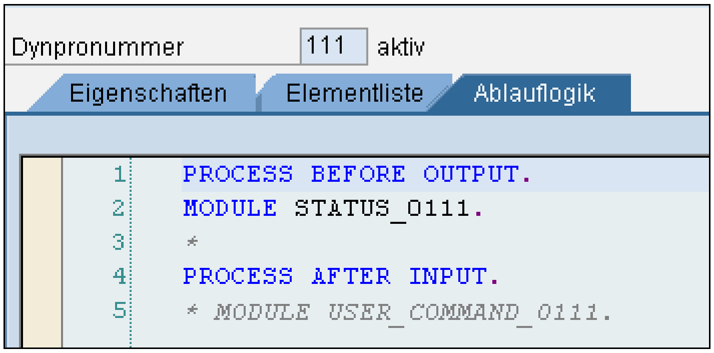
Create Dynpro SAPLXM02 111 and activate it



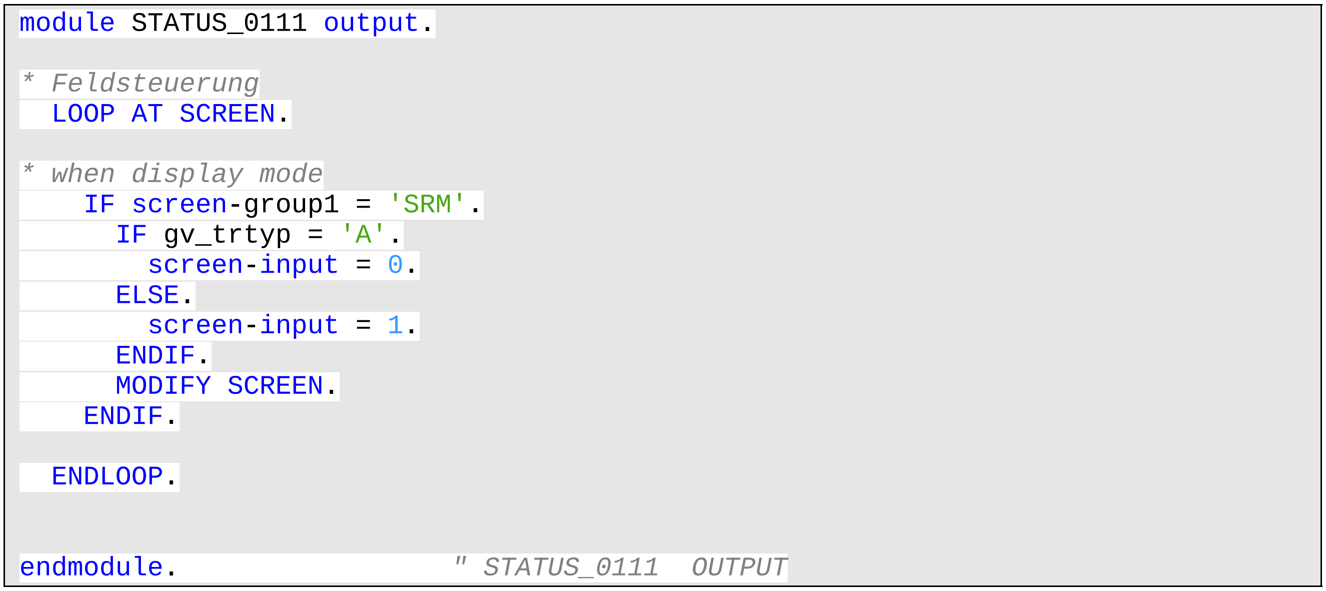
Dynpronummer	111	aktiv

H	M	Name	Typ d...	Ze...	Sp...	de...	vis...	H...	rol...	Format	Ei...	Au...	Nur...	Di...	Dict...	Property-Liste
		%#AUTOTEXT001	Frame	3	4	40	40	6			□	□	□	□		Properties
		%#AUTOTEXT002	Text	5	7	19	19	1			□	□	□	□		Properties
		EBAN-ZZ_BUDGETFLAG	Check	5	28	1	1	1		CHAR	☑	☑	□	☑		Properties
		EBAN-ZZ_BUDGET	Text	7	7	12	12	1			□	□	□	☑	2	Properties
		EBAN-ZZ_BUDGET	I/O	7	28	12	12	1	□	CHAR	☑	☑	□	☑		Properties
			OK	0	0	20	20	1	□	OK				□		

create PBO Modul

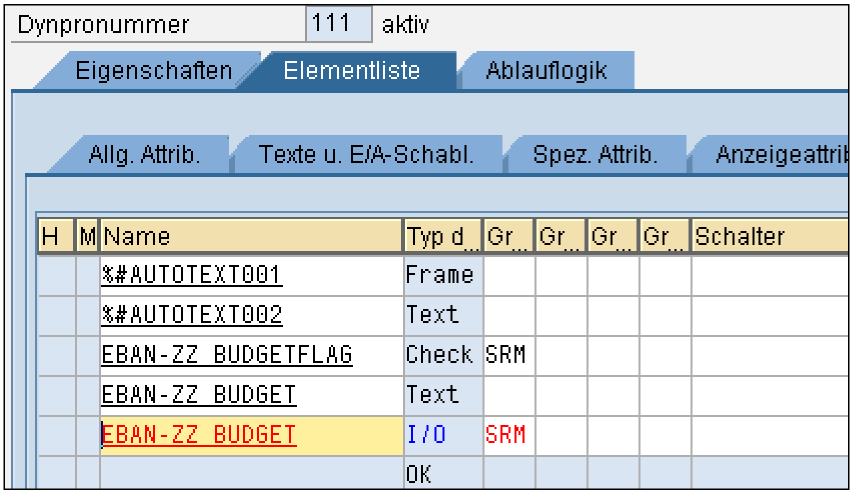


```
PROCESS BEFORE OUTPUT.
MODULE STATUS_0111.
*
PROCESS AFTER INPUT.
* MODULE USER_COMMAND_0111.
```



```
module STATUS_0111 output.

* Feldsteuerung
  LOOP AT SCREEN.

* when display mode
    IF screen-group1 = 'SRM'.
      IF gv_trtyp = 'A'.
        screen-input = 0.
      ELSE.
        screen-input = 1.
      ENDIF.
      MODIFY SCREEN.
    ENDIF.

  ENDLOOP.


endmodule.                           " STATUS_0111  OUTPUT
```

dynpro's modification group



H	M	Name	Typ d...	Gr...	Gr...	Gr...	Gr...	Schalter
		%#AUTOTEXT001	Frame					
		%#AUTOTEXT002	Text					
		EBAN-ZZ_BUDGETFLAG	Check	SRM				
		EBAN-ZZ_BUDGET	Text					
		EBAN-ZZ_BUDGET	I/O	SRM				
			OK					

In SAP-Enhancement MEREQ001 implement the exit „EXIT_SAPLMEREQ_001“, Include „ZXM02U01“ (PBO)

Here in the Top Include (global data of the exit) ZXM02TOP the field „gv_trtyp“ is defined holding the activity mode. The mode can be used in the PBO module of dynpro 111 (pls see above).

```
*&---------------------------------------------------------------------*
*&  Include           ZXM02U01
*&---------------------------------------------------------------------*

* Schreiben aus DB in den Screen

TABLES : ci_ebandb, eban.
DATA : lv_mereq_item TYPE mereq_item.

* Welcher Modus ?
CALL METHOD im_req_item->get_activity
  RECEIVING
    re_aktvt = gv_trtyp.

IF im_req_item IS INITIAL.
  CLEAR: ci_ebandb.
ELSE.
  lv_mereq_item = im_req_item->get_data( ).
  MOVE-CORRESPONDING lv_mereq_item TO eban.

ENDIF.
```

In SAP-Enhancement MEREQ001 implement the exit „EXIT_SAPLMEREQ_003“, Include „ZXM02U03“ (PAI)

```
*&---------------------------------------------------------------------*
*&  Include           ZXM02U03
*&---------------------------------------------------------------------*

* Schreiben von Screen in DB

DATA: lv_req_item TYPE mereq_item.
DATA: re_aktvt TYPE aktvt.

CALL METHOD im_req_item->get_data
  RECEIVING
    re_data = lv_req_item.

* Welcher Modus ?
CALL METHOD im_req_item->get_activity
  RECEIVING
    re_aktvt = re_aktvt.
```

```
CASE re_aktvt.
* Display mode
  WHEN 'A'.
* Change mode
  WHEN 'V'.
    lv_req_item-zz_budget = eban-zz_budget.
    lv_req_item-zz_budgetflag = eban-zz_budgetflag.
    CALL METHOD im_req_item->set_data
      EXPORTING
        im_data = lv_req_item.
    ex_changed = 'X'.
ENDCASE.
```

When transporting to the consolidation or production system activate both enhancements

MEREQ001   und   MM06E005

Transfer PurchREQ: Implement BAdi „BBP_CREATE_BE_RQ_NEW“, Method „FILL_RQ_INTERFACE“

Tip:

In the SRM system this BAdi is called with user „WF-BATCH“. Set a breakpoint for his user in your implementation. After having sent a requisition from SRm to the backend a debugger will come up. (Perhaps you have to make the user WF-BATCH a dialog user before)

In the backend the BAPI „“ is called. You can debug it there with user „SRM_MGR“.

```
METHOD IF_EX_BBP_CREATE_BE_RQ_NEW~FILL_RQ_INTERFACE.

  DATA LT_SC_ITEM      TYPE BBPT_PD_TRANSFER_ITEM.
  DATA WA_SC_ITEM      LIKE LINE OF LT_SC_ITEM.
  DATA LT_EXTENSION    TYPE TABLE OF BAPIPAREX.
  DATA WA_EXTENSION    LIKE LINE OF LT_EXTENSION.
  DATA LV_ITEMNUMBER   TYPE BNFPO.

  LOOP AT IS_SC_DOCUMENT-ITEM INTO WA_SC_ITEM.

    CONCATENATE WA_SC_ITEM-NUMBER_INT+6(4) '0' INTO LV_ITEMNUMBER.
* Extensionstruktur füllen
    WA_EXTENSION-STRUCTURE = 'BAPI_TE_REQUISITION_ITEM'.
* Positionnumber (5)=00010, Fieldcontent of ZZ-BUDGETFLAG=X, Fieldcontent
of ZZ_BUDGET=1234567
* example WA_EXTENSION-VALUEPART1 = 00010X1234567
    CONCATENATE LV_ITEMNUMBER WA_SC_ITEM-ZZ_BUDGETFLAG WA_SC_ITEM-
ZZ_BUDGET INTO WA_EXTENSION-VALUEPART1 RESPECTING BLANKS.
    APPEND WA_EXTENSION TO LT_EXTENSION.
    CLEAR WA_EXTENSION.

    CS_RQ_DOCUMENT-IT_EXTENSIONIN = LT_EXTENSION.
  ENDLOOP.

ENDMETHOD.
```

Transfer Purchase Order: Implement BAdi „BBP_CREATE_BE_PO_NEW“, Method „FILL_PO_INTERFACE1“

Tip:

In the SRM system this BAdi is called with user „WF-BATCH“. Set a breakpoint for his user in your implementation. After having sent a requisition from SRm to the backend a debugger will come up. (Perhaps you have to make the user WF-BATCH a dialog user before)

In the backend the BAPI „BAPI_PO_CREATE1“ is called. You can debug it there with user „SRM_MGR“.

```
METHOD IF_EX_BBP_CREATE_BE_PO_NEW~FILL_PO_INTERFACE1.

  DATA LT_SC_ITEM      TYPE BBPT_PD_TRANSFER_ITEM.
  DATA WA_SC_ITEM      LIKE LINE OF LT_SC_ITEM.
  DATA LT_EXTENSION    TYPE TABLE OF BAPIPAREX.
  DATA WA_EXTENSION    LIKE LINE OF LT_EXTENSION.
  DATA LV_ITEMNUMBER   TYPE BNFPO.


  LOOP AT IS_SC_DOCUMENT-ITEM INTO WA_SC_ITEM.

    CONCATENATE WA_SC_ITEM-NUMBER_INT+6(4) '0' INTO LV_ITEMNUMBER.
* Extensionstruktur füllen
    WA_EXTENSION-STRUCTURE = 'BAPI_TE_MEPOITEM'.
* Positionnumber (5)=00010, Fieldcontent of ZZ-BUDGETFLAG=X, Fieldcontent
of ZZ_BUDGET=1234567
* example WA_EXTENSION-VALUEPART1 = 00010X1234567
    CONCATENATE LV_ITEMNUMBER WA_SC_ITEM-ZZ_BUDGETFLAG WA_SC_ITEM-
ZZ_BUDGET INTO WA_EXTENSION-VALUEPART1 RESPECTING BLANKS.
    APPEND WA_EXTENSION TO LT_EXTENSION.
    CLEAR WA_EXTENSION.

* X-Struktur füllen – needed for PO
    WA_EXTENSION-STRUCTURE = 'BAPI_TE_MEPOITEMX'.
    CONCATENATE LV_ITEMNUMBER 'XX' INTO WA_EXTENSION-
VALUEPART1 RESPECTING BLANKS.
    APPEND WA_EXTENSION TO LT_EXTENSION.
    CLEAR WA_EXTENSION.

    CS_PO1_DOCUMENT-IT_EXTENSIONIN = LT_EXTENSION.

  ENDLOOP.

ENDMETHOD.
```

Copyright