SAP Fiori Notifications – Part 1

14. Januar 2020 von *Igor Muntoreanu*

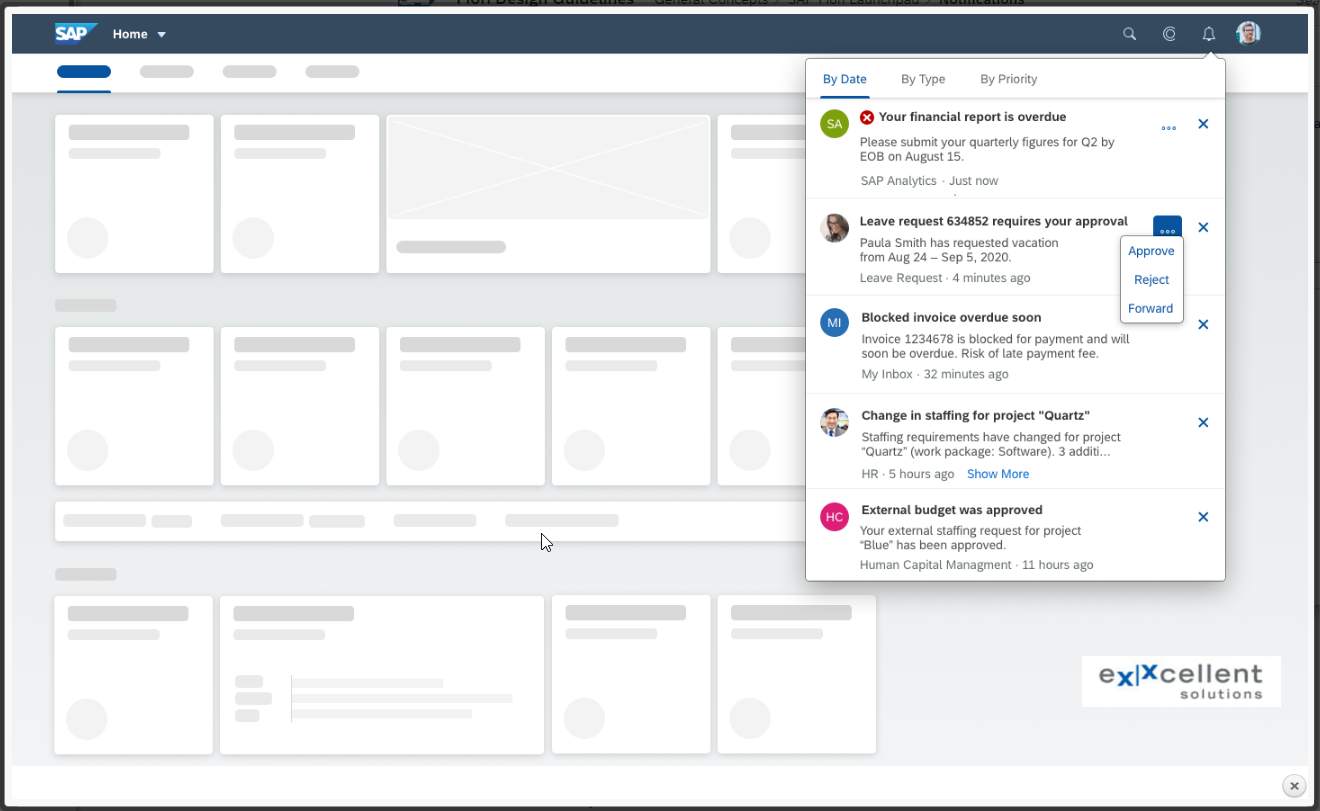
In this blog we will explain how to implement SAP Fiori Notifications on your system and provide some real Business Scenarios with the code snippets. So we start with the questions what are SAP Fiori Notifications?

Notifications in enterprise applications are an effective way to make users aware of situations that require their timely action or attention. Examples include workflow scenarios, such as the assignment of new purchase order approvals or a situation that has just arisen.

From Fiori 2.0, SAP Fiori Launchpad provides a Notifications area, by clicking the bell icon in the shell bar at the top right of the screen, that lets you know about important tasks and requests requiring your timely action or knowledge.

In the notifications popover, the user can order notifications in various ways, take action, and navigate to the source of a specific notification.

SAP Fiori Launchpad can consume notifications from the SAP Gateway Notification Channel, which is part of the SAP\_GWFND software component. The SAP Gateway Notification Channel is a framework for applications to deliver notifications to end users through various channels.

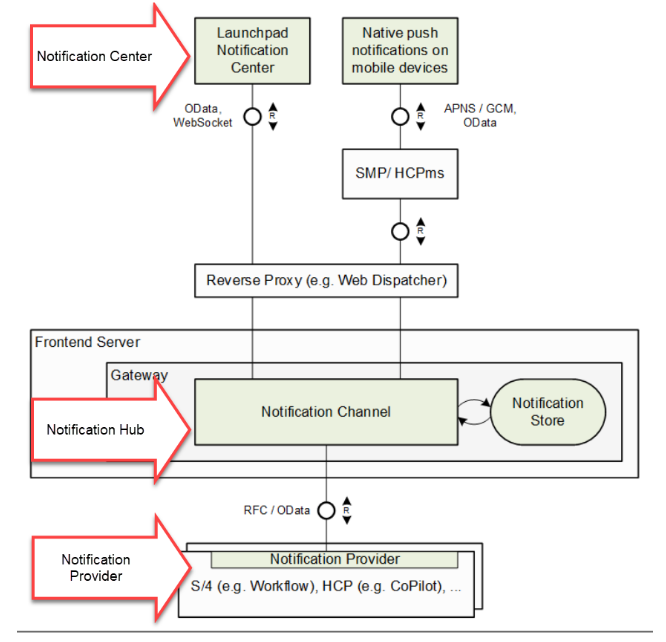


Prerequisites

The following prerequisites are required to get full usage of the Fiori Notifications:

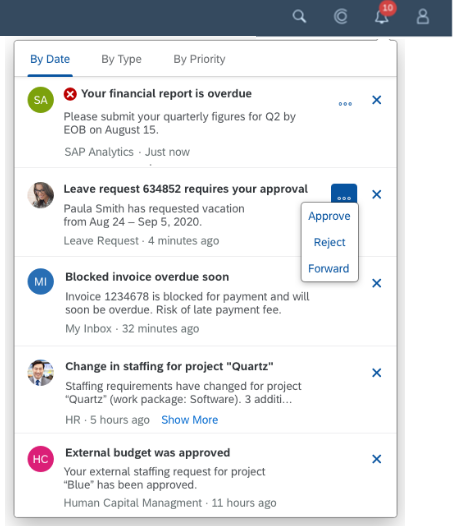
* The component SAP\_UI with version 751 SP00 and GWFND with version 7.51 SP00 or higher on the Notification Hub system.
* SAP\_BASIS with version 7.51 SP01 or higher on the Backend system(See Note: [2356153](https://launchpad.support.sap.com/#/notes/2356153) - Business Workflow Extensions - Version 2)
* Fiori Launchpad is configured (see note [2456330](https://launchpad.support.sap.com/#/notes/2456330) - How to configure Fiori Launchpad step-by-step).
* If you want to use the workflow notification, Fiori My Inbox need to be configured(see note [2424054](https://launchpad.support.sap.com/#/notes/2424054) - How to setup My Inbox 2.0 app).

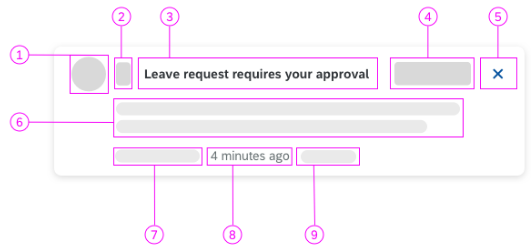
More details are described in the SAP Note: [2729492](https://launchpad.support.sap.com/#/notes/2729492).

Architecture  
  
The Launchpad consumes notifications from the notification channel, which is a Backendsystem that aggregates notifications from different notification providers configured in your environment:  
  
  
*source:* [*https://wiki.scn.sap.com/wiki/display/SAPMOB/How+to+Setup+Notifications+in+Fiori+2.0+step-by-step*](https://wiki.scn.sap.com/wiki/display/SAPMOB/How+to+Setup+Notifications+in+Fiori+2.0+step-by-step)

There are three concepts to understand when activating notifications:

* Notification Center – the area in the Fiori Launchpad where notifications are displayed.
* Notification Hub – Collects notifications to be shown in the Notification Center.
* Notification Provider – Provides notification content to the Notification Hub.

The Notifications icon button is embedded in the shell bar and is the central access point for all notifications:  
  
  
  
Below we have an example of a list of notifications and the explanation of the schematic behind it:  
  


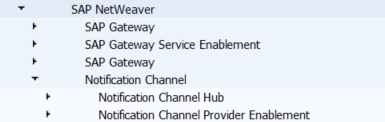


1. Image
2. Priority
3. Title
4. Action
5. Operation
6. Subtitle
7. Author/Source
8. Timestamp
9. Show/hide truncated text (Show More, Show Less)

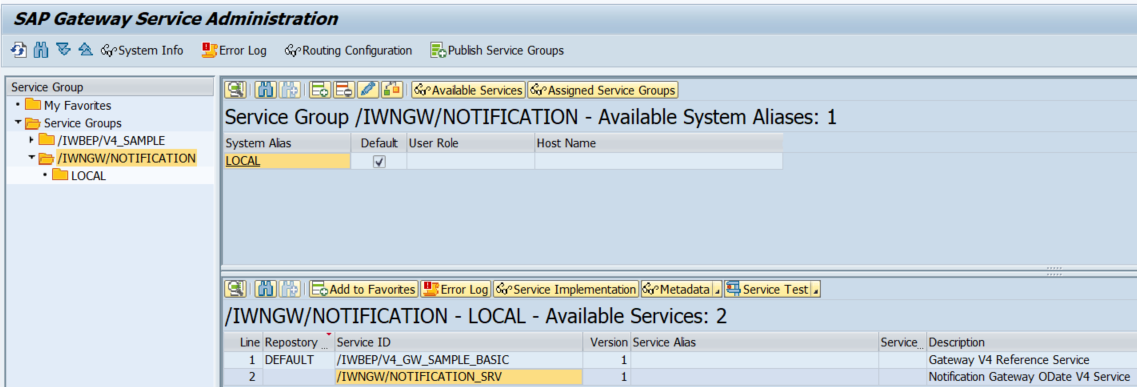
The Schematic above is an example of the version 1.84 of the SAPUI5, depending of the version, the schematic can vary, to see more details about it, check the Fiori Guidelines at the following link: <https://experience.sap.com/fiori-design-web/notification-center/> .

Implementation  
  
There are mainly 4 steps to enable the Fiori Notifications:

* Creation and assignment of certain PFCG roles.
* Configuration of the Notification Hub.
* Configuration of the Notification Providers.
* Activation of the Notifications in the Fiori Launchpad.

Mainly all the configurations are performed at the SPRO transaction, under the path:  
  
  
  
All the steps above are very well documented in the SAP support website and can vary depending of your system version. To see the documentation, please check the note 2488945 at <https://launchpad.support.sap.com/#/notes/2488945>.

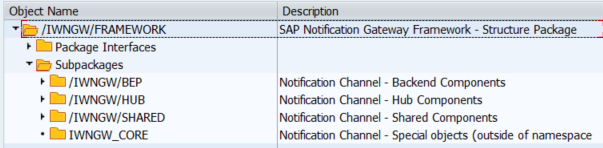
Technical Details

The notifications are collected from the Backend through the Service Id: /IWNGW/NOTIFICATION\_SRV under the Service Group /IWNGW/NOTIFICATION. It was developed using the new approach of the OData v4 and can be seen under the transaction /IWFND/V4\_ADMIN:  
  


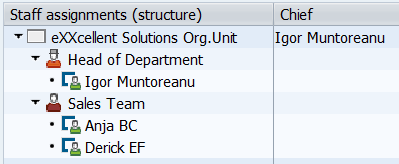
Getting even more deeply we have some database tables which handle and keep all the fiori notifications:

* /IWNGW/NOTIF - HUB Transactional - Notification Instance
* /IWNGW/NRECIP - HUB Transactional - Notification Recipients
* /IWNGW/NACTION - HUB Metadata - Notification Type Action
* /IWNGW/NTYPE - HUB Metadata - Notification Type Cache
* /IWNGW/NTYPE\_TT - HUB Metadata - Notification Type Text Template

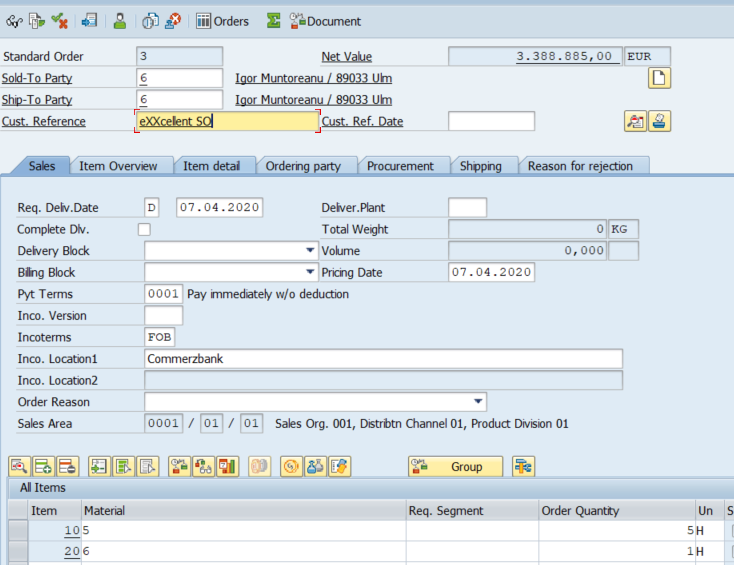
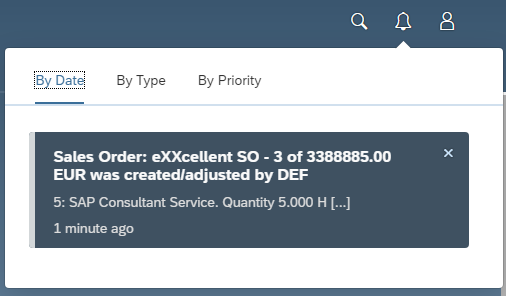
All the technical objects are stored inside the package /IWNGW/FRAMEWORK - SAP Notification Gateway Framework - Structure Package, and its subpackages:



Real Business Case Scenario 1

In this scenario we have an Organizational Unit which has a Head of Department and also employees of the Sales Team:  
  
  
  
The requirement is to send a notification to the Head of Department always when a Sales Order is adjusted or created by someone of the Sales Team.

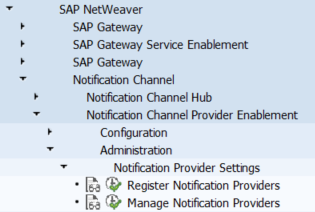
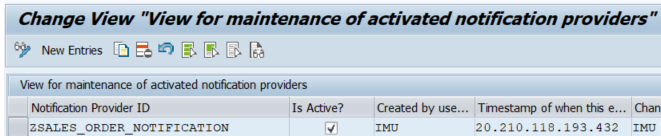
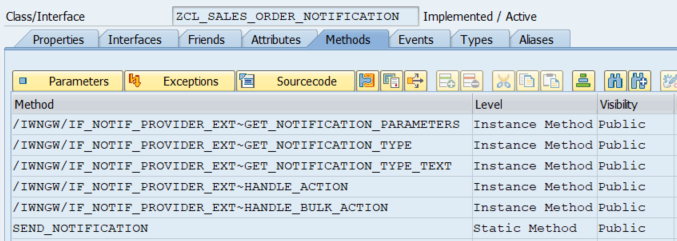
As an example the following sales order was created by Derick, sap user DEF and then the Head of Department has checked on his Fiori Launchpad the received Notification:

* Sales order created trough transaction VA01  
  
* Notification on the Fiori Lauchpad of the Head of Department:  
  

This notification is segmented in 2 parts: title and subtitle. Both have informations from the sales order. Here is how the title and subtitle were technically mapped:

|  |  |
| --- | --- |
| **Text** | **Table Source** |
| Sales Order: &1 - &2 of &3 &4 was created/adjusted by &5 | &1: VBAK-BSTNK  &2: VBAK-VBELN  &3: VBAK-NETWR  &4: VBAK-WAERK  &5: USER |
| &6: &7. Quantity &8 &9 [...] | &6: VBAP-MATNR  &7: VBAP-ARKTX  &8: VBAP-KWMENG  &9: VBAP-MEINS |

Technically, to implement this requirement the following 3 steps were implemented:

1. Configuration of the Notification Provider through the SPRO transaction:  
     
     
     
     
     
   
2. Implementation of the class ZCL\_SALES\_ORDER\_NOTIFICATION:
   1. There are 2 ways to connect the class with the notification provider. One of them is to make the class /IWNGW/CL\_NOTIF\_PROVIDER\_ABS as super class, this way implementing the interface /IWNGW/IF\_NOTIF\_PROVIDER\_EXT or to just by assigning the class /IWNGW/IF\_NOTIF\_PROVIDER to the class. In this example we have done the first way:  
      
   2. Method /IWNGW/IF\_NOTIF\_PROVIDER\_EXT~GET\_NOTIFICATION\_TYPE:

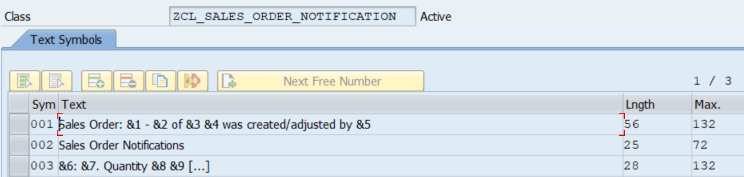
|  |
| --- |
| METHOD /iwngw/if\_notif\_provider\_ext~get\_notification\_type.    es\_notification\_type-is\_groupable = abap\_true.   es\_notification\_type-type\_key     = iv\_type\_key.   es\_notification\_type-version      = iv\_type\_version.  ENDMETHOD. |

* 1. Method /IWNGW/IF\_NOTIF\_PROVIDER\_EXT~GET\_NOTIFICATION\_TYPE\_TEXT

|  |
| --- |
| METHOD /iwngw/if\_notif\_provider\_ext~get\_notification\_type\_text.    DATA: lv\_text       TYPE string,         lv\_subtitle   TYPE string,         ls\_type\_text  TYPE /iwngw/if\_notif\_provider\_ext=>ty\_s\_notification\_type\_text.    lv\_text = TEXT-001. "Sales Order: XVBAK-BSTNK - VBAK-VBELN of VBAK-NETWR was created/adjusted by USER"   lv\_subtitle = TEXT-003. "XVBAP-MATNR: XVBAP-ARKTX. Quantity XVBAP-KWMENG XVBAP-MEINS    REPLACE '&1' WITH '{VBAK-BSTNK}'  INTO lv\_text.   REPLACE '&2' WITH '{VBAK-VBELN}'  INTO lv\_text.   REPLACE '&3' WITH '{VBAK-NETWR}'  INTO lv\_text.   REPLACE '&4' WITH '{VBAK-WAERK}'  INTO lv\_text.   REPLACE '&5' WITH '{USER}'  INTO lv\_text.   REPLACE '&6' WITH '{VBAP-MATNR}'  INTO lv\_subtitle.   REPLACE '&7' WITH '{VBAP-ARKTX}'  INTO lv\_subtitle.   REPLACE '&8' WITH '{VBAP-KWMENG}' INTO lv\_subtitle.   REPLACE '&9' WITH '{VBAP-MEINS}'  INTO lv\_subtitle.    ls\_type\_text-name  = /iwngw/if\_notif\_provider\_ext=>gc\_template\_names-template\_public.   ls\_type\_text-value = lv\_text.   APPEND ls\_type\_text TO et\_type\_text.    ls\_type\_text-name  = /iwngw/if\_notif\_provider\_ext=>gc\_template\_names-template\_sensitive.   ls\_type\_text-value = lv\_text.   APPEND ls\_type\_text TO et\_type\_text.    ls\_type\_text-name  = /iwngw/if\_notif\_provider\_ext=>gc\_template\_names-template\_grouped.   ls\_type\_text-value = TEXT-002. "Sales Order Notifications   APPEND ls\_type\_text TO et\_type\_text.    ls\_type\_text-name  = /iwngw/if\_notif\_provider\_ext=>gc\_template\_names-subtitle.   ls\_type\_text-value = lv\_subtitle. "Sales Order Notifications   APPEND ls\_type\_text TO et\_type\_text.  ENDMETHOD. |

* 1. Method SEND\_NOTIFICATION

|  |
| --- |
| METHOD send\_notification.    DATA: lt\_nf               TYPE /iwngw/if\_notif\_provider=>ty\_t\_notification,         ls\_nf               TYPE LINE OF /iwngw/if\_notif\_provider=>ty\_t\_notification,         lt\_user             TYPE STANDARD TABLE OF /iwngw/if\_notif\_provider=>ty\_s\_notification\_recipient,         ls\_user             TYPE /iwngw/if\_notif\_provider=>ty\_s\_notification\_recipient,         lt\_parameter\_values TYPE /iwngw/if\_notif\_provider=>ty\_t\_notification\_parameter,         ls\_parameter\_values TYPE LINE OF /iwngw/if\_notif\_provider=>ty\_t\_notification\_parameter,         ls\_parameter        TYPE LINE OF /iwngw/if\_notif\_provider=>ty\_t\_notification\_param\_bundle,         lt\_parameter        TYPE /iwngw/if\_notif\_provider=>ty\_t\_notification\_param\_bundle,         lt\_objec            TYPE STANDARD TABLE OF objec.    FIELD-SYMBOLS: <fs\_value> TYPE any.    "Get everyone from the Organization Unit   CALL FUNCTION 'RH\_STRUC\_GET'     EXPORTING       act\_otype      = 'O'       act\_objid      = '50000077' " eXXcellent Solutions Organization Unit       act\_wegid      = 'AI\_ORGUS' " All users of the Organization Unit     TABLES       result\_objec   = lt\_objec     EXCEPTIONS       no\_plvar\_found = 1       no\_entry\_found = 2       OTHERS         = 3.   IF sy-subrc = 0.     "Send notification only to the Head of Department     READ TABLE lt\_objec TRANSPORTING NO FIELDS WITH KEY objid = '50000079'.     IF sy-subrc = 0.       DATA(lv\_tabix) = sy-tabix + 1.       TRY.           DATA(ls\_object) = lt\_objec[ lv\_tabix ].            CHECK sy-uname <> ls\_object-realo. "Avoid the Head of sending the notification to himself            READ TABLE lt\_objec TRANSPORTING NO FIELDS WITH KEY realo = sy-uname. "Make sure that the user is in the Organization Unit           IF sy-subrc = 0.              TRY.                 DATA(lv\_system\_uuid) = cl\_uuid\_factory=>create\_system\_uuid( ).                 ls\_nf-id             = lv\_system\_uuid->create\_uuid\_x16( ).                  ls\_user-id = ls\_object-realo.                 APPEND ls\_user TO lt\_user. CLEAR ls\_user.                  ls\_nf-type\_key                 = 'ZSALES\_ORDER\_NOTIFICATION'.                 ls\_nf-type\_version             = '9'.                 ls\_nf-recipients               = lt\_user[].                  ls\_parameter\_values-name         = 'USER'.                 ls\_parameter\_values-type         = 'Edm.String'.                 ls\_parameter\_values-value        = sy-uname.                 ls\_parameter\_values-is\_sensitive = abap\_false.                 APPEND ls\_parameter\_values TO lt\_parameter\_values. CLEAR: ls\_parameter\_values.                  ls\_parameter\_values-name         = 'VBAK-VBELN'.                 ls\_parameter\_values-type         = 'Edm.String'.                 ASSIGN COMPONENT 'VBELN' OF STRUCTURE is\_xvbak TO <fs\_value>.                 ls\_parameter\_values-value        = <fs\_value>.                 SHIFT ls\_parameter\_values-value LEFT DELETING LEADING '0'.                 ls\_parameter\_values-is\_sensitive = abap\_false.                 APPEND ls\_parameter\_values TO lt\_parameter\_values. CLEAR: ls\_parameter\_values.                  ls\_parameter\_values-name         = 'VBAK-NETWR'.                 ls\_parameter\_values-type         = 'Edm.String'.                 ASSIGN COMPONENT 'NETWR' OF STRUCTURE is\_xvbak TO <fs\_value>.                 ls\_parameter\_values-value        = <fs\_value>.                 ls\_parameter\_values-is\_sensitive = abap\_false.                 APPEND ls\_parameter\_values TO lt\_parameter\_values. CLEAR: ls\_parameter\_values.                  ls\_parameter\_values-name         = 'VBAK-WAERK'.                 ls\_parameter\_values-type         = 'Edm.String'.                 ASSIGN COMPONENT 'WAERK' OF STRUCTURE is\_xvbak TO <fs\_value>.                 ls\_parameter\_values-value        = <fs\_value>.                 ls\_parameter\_values-is\_sensitive = abap\_false.                 APPEND ls\_parameter\_values TO lt\_parameter\_values. CLEAR: ls\_parameter\_values.                  ls\_parameter\_values-name         = 'VBAK-BSTNK'.                 ls\_parameter\_values-type         = 'Edm.String'.                 ASSIGN COMPONENT 'BSTNK' OF STRUCTURE is\_xvbak TO <fs\_value>.                 ls\_parameter\_values-value        = <fs\_value>.                 ls\_parameter\_values-is\_sensitive = abap\_false.                 APPEND ls\_parameter\_values TO lt\_parameter\_values. CLEAR: ls\_parameter\_values.                  "Send only the first item to the notification                 READ TABLE it\_xvbap ASSIGNING FIELD-SYMBOL(<fs\_xvbap>) INDEX 1.                 IF sy-subrc = 0.                   ls\_parameter\_values-name         = 'VBAP-MATNR'.                   ls\_parameter\_values-type         = 'Edm.String'.                   ls\_parameter\_values-value        = <fs\_xvbap>-matnr.                   SHIFT ls\_parameter\_values-value LEFT DELETING LEADING '0'.                   ls\_parameter\_values-is\_sensitive = abap\_false.                   APPEND ls\_parameter\_values TO lt\_parameter\_values. CLEAR: ls\_parameter\_values.                    ls\_parameter\_values-name         = 'VBAP-ARKTX'.                   ls\_parameter\_values-type         = 'Edm.String'.                   ls\_parameter\_values-value        = <fs\_xvbap>-arktx.                   ls\_parameter\_values-is\_sensitive = abap\_false.                   APPEND ls\_parameter\_values TO lt\_parameter\_values. CLEAR: ls\_parameter\_values.                    ls\_parameter\_values-name         = 'VBAP-KWMENG'.                   ls\_parameter\_values-type         = 'Edm.String'.                   ls\_parameter\_values-value        = <fs\_xvbap>-kwmeng.                   ls\_parameter\_values-is\_sensitive = abap\_false.                   APPEND ls\_parameter\_values TO lt\_parameter\_values. CLEAR: ls\_parameter\_values.                    ls\_parameter\_values-name         = 'VBAP-MEINS'.                   ls\_parameter\_values-type         = 'Edm.String'.                   ls\_parameter\_values-value        = <fs\_xvbap>-meins.                   ls\_parameter\_values-is\_sensitive = abap\_false.                   APPEND ls\_parameter\_values TO lt\_parameter\_values. CLEAR: ls\_parameter\_values.                 ENDIF.                  ls\_parameter-language            = sy-langu.                 ls\_parameter-parameters          = lt\_parameter\_values.                 APPEND ls\_parameter TO lt\_parameter.                  ls\_nf-parameters = lt\_parameter[].                  APPEND ls\_nf TO lt\_nf.                  TRY.                     /iwngw/cl\_notification\_api=>create\_notifications(                       EXPORTING                         iv\_provider\_id  = 'ZSALES\_ORDER\_NOTIFICATION' " class ZCL\_GENERIC\_NF\_ANSWER                         it\_notification = lt\_nf ).                      COMMIT WORK AND WAIT.                    CATCH /iwngw/cx\_notification\_api INTO DATA(lrx\_api).                     EXIT.                 ENDTRY.                CATCH cx\_uuid\_error.                 EXIT.             ENDTRY.           ENDIF.         CATCH cx\_sy\_itab\_line\_not\_found.       ENDTRY.     ENDIF.   ENDIF.  ENDMETHOD. |

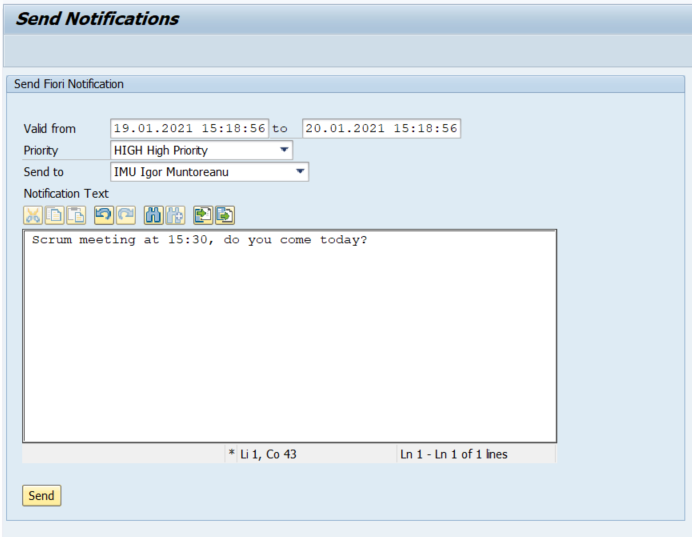
* 1. Text Elements  
       
     

1. Implementation of the user exit “userexit\_save\_document“ at “mv45afzz“

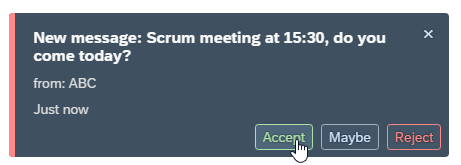
|  |
| --- |
| FORM userexit\_save\_document.    zcl\_sales\_order\_notification=>send\_notification( EXPORTING is\_xvbak = xvbak                                                              it\_xvbap = xvbap[] ).  ENDFORM. |

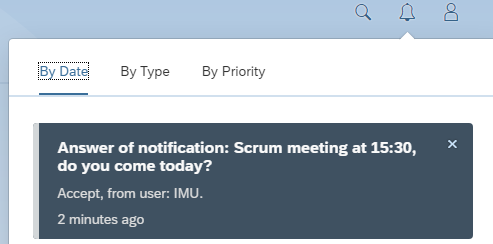
Real Business Case Scenario 2

A requirement has arrived to send simple generic fiori notifications to certain users through an abap report, within specified date and time and its validity and priority. The user will receive the notification and will have 3 possiblities of answers: Accept, Maybe and Reject. As soon as the user answers the notification, a new notification will be sent to inform the owner of the users answer. Example:



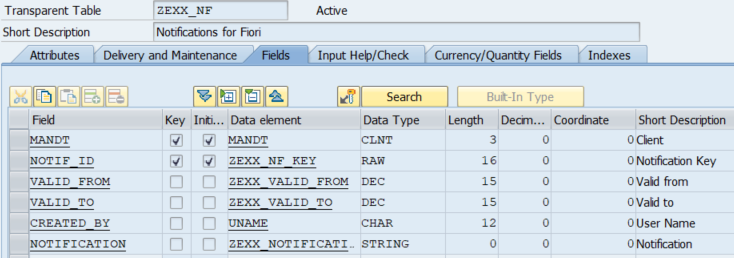
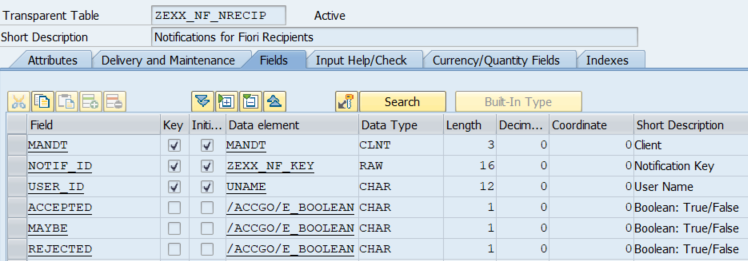
Notification received on the Fiori Launchpad for the user IMU:



When he accepts the invitation, an action will be raised and will deliver the message back to the owner of the notification:  
  


In the Abap Report, the user can plan the notification to be sent in the future, then a job is scheduled to send the notification at the specified time, that’s why have two extra tables to manage this behavior: ZEXX\_NF and ZEXX\_NF\_NRECIP.

The following 5 steps were performed to implement this requirement:

1. Creation of 2 tables
   1. ZEXX\_NF - Notifications for Fiori  
      
   2. ZEXX\_NF\_NRECIP - Notifications for Fiori Recipients  
      
2. Report ZEXX\_NOTIFICATION
   1. Body of the Report:

|  |
| --- |
| \*&---------------------------------------------------------------------\* \*& Report ZEXX\_NOTIFICATION \*&---------------------------------------------------------------------\* \*& \*&---------------------------------------------------------------------\* REPORT zexx\_notification.  INCLUDE zexx\_notification\_status\_top. INCLUDE zexx\_notification\_status\_pbo. INCLUDE zexx\_notification\_status\_pai. INCLUDE zexx\_notification\_status\_f01.  INITIALIZATION.   "check authorization   PERFORM f\_check\_authorization.    "check if provider active   PERFORM f\_check\_provider.  START-OF-SELECTION.    CALL SCREEN 0001. |

* 1. Include TOP: zexx\_notification\_status\_top

|  |
| --- |
| \*&---------------------------------------------------------------------\* \*& Include          ZEXX\_NOTIFICATION\_STATUS\_TOP \*&---------------------------------------------------------------------\*    TYPE-POOLS: vrm.   "important: it must respect the size wordwrap\_position          = 55 - check pbo   TYPES: ty\_line\_size TYPE c LENGTH 55.    DATA: \*   reference to wrapper class of control     gv\_editor           TYPE REF TO cl\_gui\_textedit,     gv\_ok\_code          TYPE sy-ucomm, \*   reference to custom container: necessary to bind TextEdit Control     gv\_editor\_container TYPE REF TO cl\_gui\_custom\_container,     gt\_mytable          TYPE STANDARD TABLE OF ty\_line\_size,     gv\_date\_from        TYPE cpet\_validfrom,     gv\_date\_to          TYPE cpet\_validfrom,     gv\_priority         TYPE /iwngw/notification\_priority.  \*   List box   DATA: gv\_field   TYPE vrm\_id,         gt\_listbox TYPE vrm\_values,         gs\_listbox TYPE LINE OF vrm\_values,         gs\_user    TYPE uname.  \*   Bapi User   DATA: gs\_address TYPE bapiaddr3,         gt\_return  TYPE STANDARD TABLE OF bapiret2. |

* 1. Include PBO: zexx\_notification\_status\_pbo

|  |
| --- |
| \*&---------------------------------------------------------------------\* \*& Include          ZEXX\_NOTIFICATION\_STATUS\_PBO \*&---------------------------------------------------------------------\* \*&---------------------------------------------------------------------\* \*&      Module  STATUS\_0001  OUTPUT \*&---------------------------------------------------------------------\* \*       text \*----------------------------------------------------------------------\* MODULE status\_0001 OUTPUT.    DATA: lv\_morgen TYPE sy-datum.    SET PF-STATUS 'MAIN100'.   SET TITLEBAR '0001'.    IF gv\_date\_from IS INITIAL.     gv\_date\_from = sy-datlo && sy-timlo.   ENDIF.    IF gv\_date\_to IS INITIAL.     lv\_morgen = sy-datlo + 1.     gv\_date\_to = lv\_morgen && sy-timlo.   ENDIF.    IF gv\_editor IS INITIAL.  \*   create control container     CREATE OBJECT gv\_editor\_container       EXPORTING         container\_name              = 'GV\_TEXT\_EDITOR'       EXCEPTIONS         cntl\_error                  = 1         cntl\_system\_error           = 2         create\_error                = 3         lifetime\_error              = 4         lifetime\_dynpro\_dynpro\_link = 5.     IF sy-subrc EQ 0.  \*   create calls constructor, which initializes, creats and links \*   TextEdit Control       CREATE OBJECT gv\_editor         EXPORTING           parent                     = gv\_editor\_container           wordwrap\_mode              = cl\_gui\_textedit=>wordwrap\_at\_fixed\_position           wordwrap\_position          = 55           wordwrap\_to\_linebreak\_mode = cl\_gui\_textedit=>true.        REFRESH gt\_mytable.  " to initialize table upon OK\_CODE 'BACK' at PAI      ENDIF.   ENDIF.    "Users Listbox:   REFRESH gt\_listbox.   SELECT DISTINCT uname          FROM agr\_users          INTO TABLE @DATA(gt\_users)          WHERE agr\_name = 'ZNTUSERPRODU'             OR agr\_name = 'ZNTUSERCONSU'             OR agr\_name = 'ZNTNPDEV'             OR agr\_name = 'ZNTHUBADM'             OR agr\_name = 'ZNTBEPADM'             AND from\_dat <= @sy-datum             AND to\_dat >= @sy-datum.   IF sy-subrc = 0.     LOOP AT gt\_users INTO DATA(ls\_users).       gs\_listbox-key  = ls\_users-uname.        CALL FUNCTION 'BAPI\_USER\_GET\_DETAIL'         EXPORTING           username = ls\_users-uname         IMPORTING           address  = gs\_address         TABLES           return   = gt\_return.        CONCATENATE gs\_address-firstname gs\_address-lastname INTO gs\_listbox-text SEPARATED BY space.        APPEND gs\_listbox TO gt\_listbox. CLEAR gs\_listbox.     ENDLOOP.      gs\_listbox-key = '\*'.     gs\_listbox-text = text-006.     APPEND gs\_listbox TO gt\_listbox. CLEAR gs\_listbox.      gv\_field = 'GS\_USER'.      CALL FUNCTION 'VRM\_SET\_VALUES'       EXPORTING         id     = gv\_field         values = gt\_listbox.    ENDIF.  ENDMODULE. |

* 1. Include PAI: zexx\_notification\_status\_pai

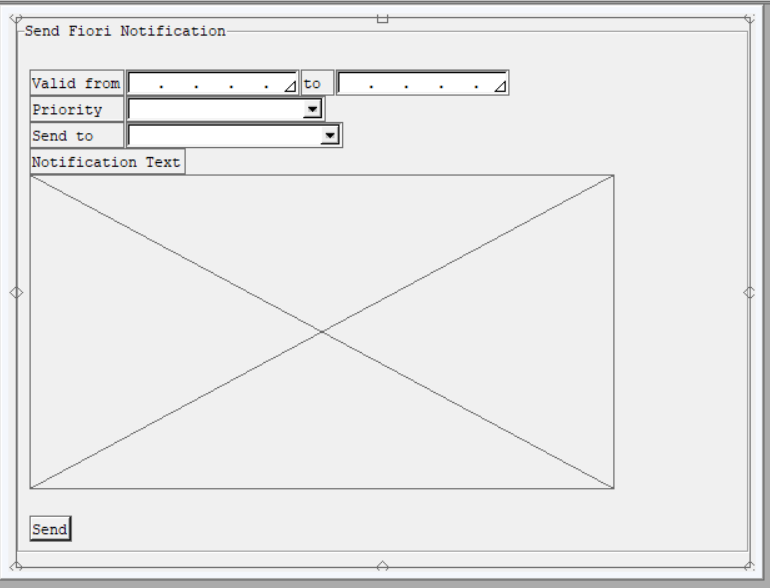
|  |
| --- |
| \*&---------------------------------------------------------------------\* \*& Include          ZEXX\_NOTIFICATION\_STATUS\_PAI \*&---------------------------------------------------------------------\* \*&---------------------------------------------------------------------\* \*&      Module  USER\_COMMAND\_0001  INPUT \*&---------------------------------------------------------------------\* MODULE user\_command\_0001 INPUT.   CASE gv\_ok\_code.     WHEN 'BACK' OR 'LEAVE' OR 'EXIT'. \* Destroy Control.       IF NOT gv\_editor IS INITIAL.         CALL METHOD gv\_editor->free           EXCEPTIONS             OTHERS = 1.         IF sy-subrc NE 0.           FREE gv\_editor.         ELSE. \*   free ABAP object also           FREE gv\_editor.         ENDIF.       ENDIF. \* destroy container       IF NOT gv\_editor\_container IS INITIAL.         CALL METHOD gv\_editor\_container->free           EXCEPTIONS             OTHERS = 1.         IF sy-subrc EQ 0.           FREE gv\_editor\_container.         ELSE. \*   free ABAP object also           FREE gv\_editor\_container.         ENDIF.       ENDIF.  \* finally flush       CALL METHOD cl\_gui\_cfw=>flush         EXCEPTIONS           OTHERS = 1.       IF sy-subrc NE 0.         LEAVE PROGRAM.       ENDIF.       LEAVE PROGRAM.     WHEN 'SEND'.       PERFORM f\_send\_notification.     WHEN OTHERS.   ENDCASE.    CLEAR gv\_ok\_code.  ENDMODULE. |

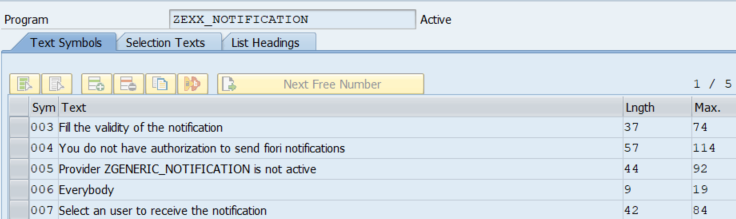
* 1. Include F01: zexx\_notification\_status\_f01

|  |
| --- |
| \*&---------------------------------------------------------------------\* \*& Include          ZEXX\_NOTIFICATION\_STATUS\_F01 \*&---------------------------------------------------------------------\* \*&---------------------------------------------------------------------\* \*&  Include           /PBK/NOTIFICATION\_STATUS\_F01 \*&---------------------------------------------------------------------\* \*&---------------------------------------------------------------------\* \*&      Form  F\_SEND\_NOTIFICATION \*&---------------------------------------------------------------------\* FORM f\_send\_notification .    DATA: ls\_notification TYPE zexx\_nf,         lt\_selection    TYPE STANDARD TABLE OF rsparams,         ls\_selection    TYPE rsparams.    IF gv\_date\_from IS INITIAL OR gv\_date\_to IS INITIAL.     MESSAGE TEXT-003 TYPE 'S' DISPLAY LIKE 'E'. "Fill the validity     EXIT.   ENDIF.    IF gs\_user IS INITIAL.     MESSAGE TEXT-007 TYPE 'S' DISPLAY LIKE 'E'. "Select an user to receive the notification     EXIT.   ELSE.     ls\_selection-selname = 'S\_USER'.     ls\_selection-kind    = 'S'. "Select Options = S     ls\_selection-option  = COND #( WHEN gs\_user = '\*' THEN 'CP' ELSE 'EQ' ).     ls\_selection-sign    = 'I'.     ls\_selection-low     = gs\_user.     APPEND ls\_selection TO lt\_selection. CLEAR ls\_selection.    ENDIF.    IF gv\_editor IS BOUND.      ls\_notification-notif\_id   = /iwngw/cl\_med\_util=>new\_guid( ).     ls\_notification-valid\_from = gv\_date\_from.     ls\_notification-valid\_to   = gv\_date\_to.     ls\_notification-created\_by = sy-uname.      CALL METHOD gv\_editor->get\_text\_as\_r3table       IMPORTING         table = gt\_mytable.      CONCATENATE LINES OF gt\_mytable INTO ls\_notification-notification SEPARATED BY cl\_abap\_char\_utilities=>cr\_lf.      INSERT zexx\_nf FROM ls\_notification.     IF sy-subrc = 0.        ls\_selection-selname = 'P\_ID'.       ls\_selection-kind    = 'P'. "Parameter = P       ls\_selection-option  = 'EQ'.       ls\_selection-sign    = 'I'.       ls\_selection-low     = ls\_notification-notif\_id.       APPEND ls\_selection TO lt\_selection. CLEAR ls\_selection.        ls\_selection-selname = 'P\_PRI'.       ls\_selection-kind    = 'P'. "Select Options = S       ls\_selection-option  = 'EQ'.       ls\_selection-sign    = 'I'.       ls\_selection-low     = gv\_priority.       APPEND ls\_selection TO lt\_selection. CLEAR ls\_selection.        COMMIT WORK AND WAIT.       SUBMIT zexx\_notification\_send WITH SELECTION-TABLE lt\_selection AND RETURN.      ENDIF.    ENDIF.  ENDFORM. \*&---------------------------------------------------------------------\* \*&      Form  F\_CHECK\_AUTHORIZATION \*&---------------------------------------------------------------------\* FORM f\_check\_authorization.    SELECT SINGLE agr\_name, uname, from\_dat, to\_dat          FROM agr\_users          INTO @DATA(ls\_users)          WHERE ( agr\_name = 'ZNTUSERPRODU' OR                  agr\_name = 'ZNTUSERCONSU' OR                  agr\_name = 'ZNTNPDEV'     OR                  agr\_name = 'ZNTHUBADM'    OR                  agr\_name = 'ZNTBEPADM' )             AND uname = @sy-uname             AND from\_dat <= @sy-datum             AND to\_dat >= @sy-datum.   IF sy-subrc <> 0.     "Sie haben keine Berechtigung zum Senden von Benachrichtigungen     MESSAGE TEXT-004 TYPE 'S' DISPLAY LIKE 'E'.     LEAVE PROGRAM.   ENDIF.  ENDFORM. \*&---------------------------------------------------------------------\* \*& Form f\_check\_provider \*&---------------------------------------------------------------------\* FORM f\_check\_provider .    DATA(ls\_pr\_config) = /iwngw/cl\_notification\_api=>get\_provider\_config( 'ZGENERIC\_NOTIFICATION' ).   IF ls\_pr\_config-active IS INITIAL.     "Provider is not active     MESSAGE TEXT-005 TYPE 'S' DISPLAY LIKE 'E'.     LEAVE PROGRAM.   ENDIF.    TRY.       /iwngw/cl\_cos\_provider\_reg\_cof=>assert\_provider\_id( 'ZGENERIC\_NOTIFICATION' ).     CATCH /iwngw/cx\_cos\_provider\_reg INTO DATA(lx\_cos\_provider\_reg).       "Provider is not active       MESSAGE TEXT-005 TYPE 'S' DISPLAY LIKE 'E'.       LEAVE PROGRAM.   ENDTRY.  ENDFORM. |

* 1. Screen 0001

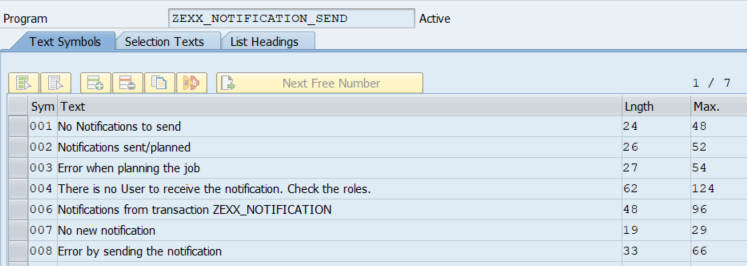
|  |
| --- |
| PROCESS BEFORE OUTPUT.   MODULE status\_0001. \* PROCESS AFTER INPUT.   MODULE user\_command\_0001. |

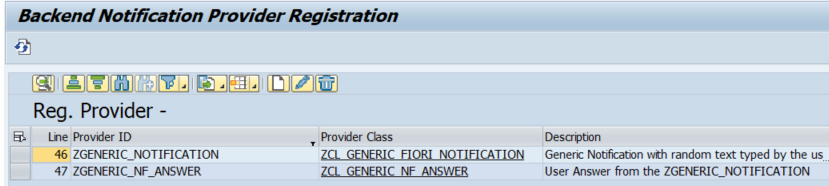
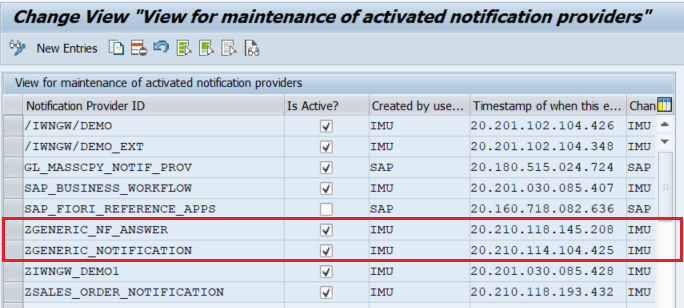


* 1. Text Elements  
       
     

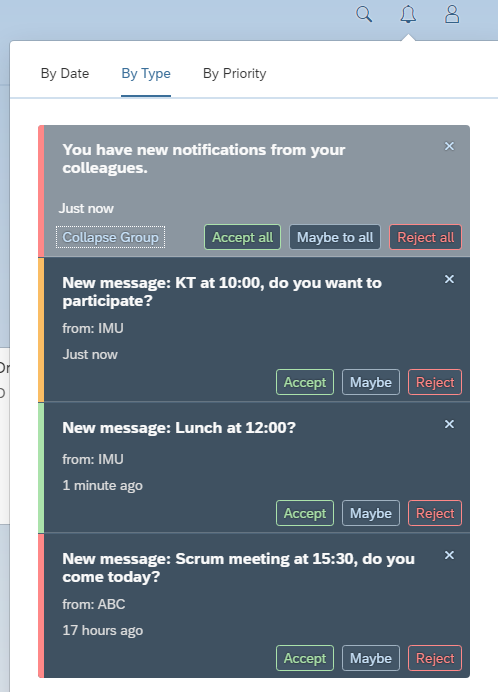
1. Report ZEXX\_NOTIFICATION\_SEND
   1. Body of the report:

|  |
| --- |
| \*&---------------------------------------------------------------------\* \*& Report ZEXX\_NOTIFICATION\_SEND \*&---------------------------------------------------------------------\* \*& This program must not be executed manually! \*  The program can be executed via job or per the main program of notifications \* /PBK/NOTIFICATION\_SEND \*&---------------------------------------------------------------------\* REPORT zexx\_notification\_send.  DATA: zexx\_nf        TYPE zexx\_nf,       zexx\_nf\_nrecip TYPE zexx\_nf\_nrecip.  TYPES: ty\_zexx\_nf        TYPE STANDARD TABLE OF zexx\_nf,        ty\_zexx\_nf\_nrecip TYPE STANDARD TABLE OF zexx\_nf\_nrecip.  DATA: gt\_users TYPE /iwngw/if\_notif\_provider=>ty\_t\_notification\_recipient.  PARAMETERS: p\_id  TYPE zexx\_nf-notif\_id,             p\_pri TYPE /iwngw/notification\_priority.  "used only for the job option SELECT-OPTIONS: s\_id   FOR zexx\_nf-notif\_id MODIF ID job,                 s\_user FOR zexx\_nf\_nrecip-user\_id.  INITIALIZATION.  AT SELECTION-SCREEN OUTPUT.   PERFORM f\_treat\_screen.  START-OF-SELECTION.    PERFORM f\_delete\_outdated.   PERFORM f\_get\_users\_to\_send\_nf.    IF sy-batch IS INITIAL.     PERFORM f\_send\_notification USING p\_id.   ELSE.     PERFORM f\_send\_notification\_batch.   ENDIF.   \*&---------------------------------------------------------------------\* \*&      Form  F\_DELETE\_OUTDATED \*&---------------------------------------------------------------------\* FORM f\_delete\_outdated.    GET TIME STAMP FIELD DATA(lv\_timestamp).   CONVERT TIME STAMP lv\_timestamp TIME ZONE sy-zonlo INTO DATE DATA(lv\_date) TIME DATA(lv\_time).   lv\_timestamp = lv\_date && lv\_time.    SELECT \*     FROM zexx\_nf     INTO TABLE @DATA(lt\_zexx\_nf)     WHERE valid\_to < @lv\_timestamp.   IF sy-subrc = 0.      SELECT notif\_id, type\_id       FROM /iwngw/notif       INTO TABLE @DATA(lt\_/iwngw/notif)       FOR ALL ENTRIES IN @lt\_zexx\_nf       WHERE notif\_id = @lt\_zexx\_nf-notif\_id.     IF sy-subrc = 0.       LOOP AT lt\_/iwngw/notif ASSIGNING FIELD-SYMBOL(<fs\_/iwngw/notif>).         DELETE FROM /iwngw/notif    WHERE notif\_id = <fs\_/iwngw/notif>-notif\_id.         DELETE FROM /iwngw/nrecip   WHERE notif\_id = <fs\_/iwngw/notif>-notif\_id.         DELETE FROM /iwngw/ntype    WHERE type\_id  = <fs\_/iwngw/notif>-type\_id.         DELETE FROM /iwngw/ntype\_tt WHERE type\_id  = <fs\_/iwngw/notif>-type\_id.       ENDLOOP.       COMMIT WORK AND WAIT.     ENDIF.      DELETE zexx\_nf FROM TABLE lt\_zexx\_nf.     IF sy-subrc = 0.       COMMIT WORK AND WAIT.     ENDIF.    ENDIF.  ENDFORM. \*&---------------------------------------------------------------------\* \*&      Form  F\_SEND\_NOTIFICATION \*&---------------------------------------------------------------------\* FORM f\_send\_notification  USING p\_id TYPE zexx\_nf-notif\_id.  \* " Service class /IWNGW/CL\_IN\_CO\_NF\_DATA   " Notification class to handle data /IWNGW/CL\_COS\_NOTIF\_DBA   " Important class /IWNGW/CL\_med\_ntype\_dba    DATA: lt\_zexx\_nf\_job     TYPE STANDARD TABLE OF zexx\_nf,         lt\_zexx\_nf\_receips TYPE STANDARD TABLE OF zexx\_nf\_nrecip.    GET TIME STAMP FIELD DATA(lv\_timestamp).   CONVERT TIME STAMP lv\_timestamp TIME ZONE sy-zonlo INTO DATE DATA(lv\_date) TIME DATA(lv\_time).   lv\_timestamp = lv\_date && lv\_time.    IF p\_id IS INITIAL.     SELECT \*       FROM zexx\_nf       INTO TABLE @DATA(lt\_zexx\_nf)       WHERE valid\_from <= @lv\_timestamp AND valid\_to >= @lv\_timestamp.   ELSE.     SELECT \*       FROM zexx\_nf       INTO TABLE @lt\_zexx\_nf       WHERE notif\_id = @p\_id.   ENDIF.    IF lt\_zexx\_nf IS NOT INITIAL.     LOOP AT lt\_zexx\_nf ASSIGNING FIELD-SYMBOL(<fs\_zexx\_nf>).        IF p\_id IS NOT INITIAL AND <fs\_zexx\_nf>-valid\_from > lv\_timestamp.         "it will be scheduled.         APPEND <fs\_zexx\_nf> TO lt\_zexx\_nf\_job.         CONTINUE.       ENDIF.        PERFORM f\_assign\_values\_nf\_standard USING <fs\_zexx\_nf>                                           CHANGING lt\_zexx\_nf\_receips.      ENDLOOP.      IF lt\_zexx\_nf\_receips IS NOT INITIAL.       MODIFY zexx\_nf\_nrecip FROM TABLE lt\_zexx\_nf\_receips.       COMMIT WORK.     ENDIF.      IF lt\_zexx\_nf\_job IS NOT INITIAL.       PERFORM f\_schedule\_job USING lt\_zexx\_nf\_job.     ENDIF.      "Notifications sent.     MESSAGE TEXT-002 TYPE 'S'.    ELSE.     "No notifications to send.     MESSAGE TEXT-001 TYPE 'S' DISPLAY LIKE 'E'.   ENDIF.  ENDFORM. \*&---------------------------------------------------------------------\* \*&      Form  F\_SCHEDULE\_JOB \*&---------------------------------------------------------------------\* FORM f\_schedule\_job  USING lt\_zexx\_nf\_job TYPE ty\_zexx\_nf.    DATA: lv\_valid     TYPE c,         ls\_params    TYPE pri\_params,         lv\_jobcount  TYPE tbtcjob-jobcount,         lv\_released  TYPE btch0000-char1,         lv\_jobname   TYPE tbtcjob-jobname,         lt\_selection TYPE STANDARD TABLE OF rsparams,         ls\_selection TYPE rsparams,         lv\_plantime  TYPE tbtcjob-sdlstrttm,         lv\_plandate  TYPE tbtcjob-sdlstrtdt.    CALL FUNCTION 'GET\_PRINT\_PARAMETERS'     EXPORTING       no\_dialog              = abap\_true     IMPORTING       out\_parameters         = ls\_params       valid                  = lv\_valid     EXCEPTIONS       archive\_info\_not\_found = 1       invalid\_print\_params   = 2       invalid\_archive\_params = 3       OTHERS                 = 4.   IF sy-subrc = 0.     lv\_jobname = sy-repid.      CALL FUNCTION 'JOB\_OPEN'       EXPORTING         jobname          = lv\_jobname       IMPORTING         jobcount         = lv\_jobcount       EXCEPTIONS         cant\_create\_job  = 1         invalid\_job\_data = 2         jobname\_missing  = 3         OTHERS           = 4.     IF sy-subrc = 0.        "We will have alway only one       LOOP AT lt\_zexx\_nf\_job INTO DATA(ls\_zexx\_nf\_job).         ls\_selection-selname = 'S\_ID'.         ls\_selection-kind    = 'S'. "Select Options = S         ls\_selection-sign    = 'I'.         ls\_selection-option  = 'EQ'.         ls\_selection-low     = ls\_zexx\_nf\_job-notif\_id.         INSERT ls\_selection INTO TABLE lt\_selection.         CLEAR ls\_selection.       ENDLOOP.        SUBMIT zexx\_notification\_send WITH SELECTION-TABLE lt\_selection       VIA JOB lv\_jobname       NUMBER lv\_jobcount       TO SAP-SPOOL WITHOUT SPOOL DYNPRO       SPOOL PARAMETERS ls\_params       AND RETURN.        "Job will be scheduled according to the System Time Zone       CONVERT TIME STAMP ls\_zexx\_nf\_job-valid\_from TIME ZONE 'UTC-1' INTO DATE lv\_plandate TIME lv\_plantime.        CALL FUNCTION 'JOB\_CLOSE'         EXPORTING           jobcount             = lv\_jobcount           jobname              = lv\_jobname           sdlstrtdt            = lv\_plandate           sdlstrttm            = lv\_plantime           strtimmed            = abap\_true         IMPORTING           job\_was\_released     = lv\_released         EXCEPTIONS           cant\_start\_immediate = 1           invalid\_startdate    = 2           jobname\_missing      = 3           job\_close\_failed     = 4           job\_nosteps          = 5           job\_notex            = 6           lock\_failed          = 7           invalid\_target       = 8           OTHERS               = 9.        IF sy-subrc NE 0.         MESSAGE TEXT-003 TYPE 'E'. "error       ENDIF.     ENDIF.   ENDIF.  ENDFORM. \*&---------------------------------------------------------------------\* \*&      Form  F\_SEND\_NOTIFICATION\_BATCH \*&---------------------------------------------------------------------\* FORM f\_send\_notification\_batch.    DATA: lt\_zexx\_nf\_receips TYPE STANDARD TABLE OF zexx\_nf\_nrecip.    IF s\_id IS NOT INITIAL.     SELECT \*       FROM zexx\_nf       INTO TABLE @DATA(lt\_zexx\_nf)       WHERE notif\_id IN @s\_id.      IF lt\_zexx\_nf IS NOT INITIAL.       LOOP AT lt\_zexx\_nf ASSIGNING FIELD-SYMBOL(<fs\_zexx\_nf>).          PERFORM f\_assign\_values\_nf\_standard USING <fs\_zexx\_nf>                                             CHANGING lt\_zexx\_nf\_receips.          IF lt\_zexx\_nf\_receips IS NOT INITIAL.           MODIFY zexx\_nf\_nrecip FROM TABLE lt\_zexx\_nf\_receips.           COMMIT WORK.         ENDIF.        ENDLOOP.        "Notifications sent.       MESSAGE TEXT-002 TYPE 'S'.        LOOP AT lt\_zexx\_nf ASSIGNING <fs\_zexx\_nf>.         WRITE: / <fs\_zexx\_nf>-notif\_id, space, <fs\_zexx\_nf>-valid\_from, space , <fs\_zexx\_nf>-valid\_to, space, <fs\_zexx\_nf>-notification.       ENDLOOP.      ELSE.       "No notifications to send.       MESSAGE TEXT-001 TYPE 'S' DISPLAY LIKE 'E'.     ENDIF.    ELSE.     "No new Notifications     MESSAGE TEXT-007 TYPE 'S'.   ENDIF.  ENDFORM. \*&---------------------------------------------------------------------\* \*&      Form  F\_ASSIGN\_VALUES\_NF\_STANDARD \*&---------------------------------------------------------------------\* FORM f\_assign\_values\_nf\_standard  USING    ls\_zexx\_nf        TYPE zexx\_nf                                   CHANGING lt\_zexx\_nf\_nrecip TYPE ty\_zexx\_nf\_nrecip.    DATA: lt\_nf               TYPE /iwngw/if\_notif\_provider=>ty\_t\_notification,         lt\_parameter        TYPE /iwngw/if\_notif\_provider=>ty\_t\_notification\_param\_bundle,         lt\_parameter\_values TYPE /iwngw/if\_notif\_provider=>ty\_t\_notification\_parameter,         ls\_nf               TYPE LINE OF /iwngw/if\_notif\_provider=>ty\_t\_notification,         ls\_parameter        TYPE LINE OF /iwngw/if\_notif\_provider=>ty\_t\_notification\_param\_bundle,         ls\_parameter\_values TYPE LINE OF /iwngw/if\_notif\_provider=>ty\_t\_notification\_parameter,         lv\_system\_uuid      TYPE REF TO if\_system\_uuid.    DATA: ls\_zexx\_nf\_receips TYPE zexx\_nf\_nrecip.    zcl\_generic\_fiori\_notification=>gv\_generic\_nf = ls\_zexx\_nf-notification.    lv\_system\_uuid = cl\_uuid\_factory=>create\_system\_uuid( ).    TRY.       ls\_nf-id                       = ls\_zexx\_nf-notif\_id. "lv\_system\_uuid->create\_uuid\_x16( ).     CATCH cx\_uuid\_error.   ENDTRY.    ls\_nf-type\_key                 = 'ZGENERIC\_NOTIFICATION'.   ls\_nf-type\_version             = '9'.   ls\_nf-recipients               = gt\_users[].   ls\_nf-priority                 = p\_pri.    ls\_parameter\_values-name         = 'message'. "Check Class and Method: ZCL\_GENERIC\_FIORI\_NOTIFICATION~/IWNGW/IF\_NOTIF\_PROVIDER~GET\_NOTIFICATION\_TYPE\_TEXT   ls\_parameter\_values-type         = 'Edm.String'.   ls\_parameter\_values-value        = ls\_zexx\_nf-notification.   ls\_parameter\_values-is\_sensitive = abap\_false.   APPEND ls\_parameter\_values TO lt\_parameter\_values. CLEAR: ls\_parameter\_values.    ls\_parameter\_values-name         = 'user'. "Check Class and Method: ZCL\_GENERIC\_FIORI\_NOTIFICATION~/IWNGW/IF\_NOTIF\_PROVIDER~GET\_NOTIFICATION\_TYPE\_TEXT   ls\_parameter\_values-type         = 'Edm.String'.   ls\_parameter\_values-value        = sy-uname.   ls\_parameter\_values-is\_sensitive = abap\_false.   APPEND ls\_parameter\_values TO lt\_parameter\_values. CLEAR: ls\_parameter\_values.    ls\_parameter-language            = sy-langu.   ls\_parameter-parameters          = lt\_parameter\_values.   APPEND ls\_parameter TO lt\_parameter.    ls\_nf-parameters = lt\_parameter[].    APPEND ls\_nf TO lt\_nf.    TRY.        /iwngw/cl\_notification\_api=>create\_notifications(         EXPORTING           iv\_provider\_id  = 'ZGENERIC\_NOTIFICATION'           it\_notification = lt\_nf ).        LOOP AT gt\_users INTO DATA(ls\_user).         ls\_zexx\_nf\_receips-notif\_id = ls\_nf-id.         ls\_zexx\_nf\_receips-user\_id  = ls\_user-id.         APPEND ls\_zexx\_nf\_receips TO lt\_zexx\_nf\_nrecip. CLEAR ls\_zexx\_nf\_receips.       ENDLOOP.      CATCH /iwngw/cx\_notification\_api INTO DATA(lrx\_api).       MESSAGE TEXT-008 TYPE 'S' DISPLAY LIKE 'E'.       LEAVE LIST-PROCESSING.   ENDTRY.  ENDFORM. \*&---------------------------------------------------------------------\* \*&      Form  F\_TREAT\_SCREEN \*&---------------------------------------------------------------------\* FORM f\_treat\_screen .   LOOP AT SCREEN.     IF screen-group1 = 'JOB'.       screen-invisible = 1.       screen-active = 0.       MODIFY SCREEN.     ENDIF.   ENDLOOP. ENDFORM. \*&---------------------------------------------------------------------\* \*&      Form  F\_GET\_USERS\_TO\_SEND\_NF \*&---------------------------------------------------------------------\* FORM f\_get\_users\_to\_send\_nf .    SELECT DISTINCT uname          FROM agr\_users          INTO TABLE gt\_users          WHERE ( agr\_name = 'ZNTUSERPRODU'             OR agr\_name = 'ZNTUSERCONSU'             OR agr\_name = 'ZNTNPDEV'             OR agr\_name = 'ZNTHUBADM'             OR agr\_name = 'ZNTBEPADM' )            AND from\_dat <= sy-datum             AND to\_dat >= sy-datum             AND uname IN s\_user.   IF sy-subrc <> 0.     "There are no users to receive notifications. Check the roles.     MESSAGE TEXT-004 TYPE 'S' DISPLAY LIKE 'E'.     LEAVE LIST-PROCESSING.   ENDIF.  ENDFORM. |

* 1. Text elements:  
       
     

1. Provider Configuration at SPRO transaction  
     
     
     
   
2. Class ZCL\_GENERIC\_FIORI\_NOTIFICATION
   1. Assign the interface /IWNGW/IF\_NOTIF\_PROVIDER to the class
   2. Method /IWNGW/IF\_NOTIF\_PROVIDER~HANDLE\_ACTION: This method will perform the logic after the user decides if he will accept the question of the notification or not.

|  |
| --- |
| METHOD /iwngw/if\_notif\_provider~handle\_action.    DATA: ls\_zexx\_nf\_nrecip TYPE zexx\_nf\_nrecip.    "Update the DB according with the button the user has clicked   CASE iv\_action\_key.     WHEN 'Accept'.       SELECT SINGLE \* FROM zexx\_nf\_nrecip INTO ls\_zexx\_nf\_nrecip WHERE notif\_id = iv\_notification\_id AND user\_id = sy-uname.       IF sy-subrc = 0.         ls\_zexx\_nf\_nrecip-accepted = abap\_true.         UPDATE zexx\_nf\_nrecip FROM ls\_zexx\_nf\_nrecip.       ENDIF.        send\_nf\_answer( EXPORTING iv\_nf\_id = iv\_notification\_id                                 iv\_action\_key = iv\_action\_key ).      WHEN 'Reject'.       SELECT SINGLE \* FROM zexx\_nf\_nrecip INTO ls\_zexx\_nf\_nrecip WHERE notif\_id = iv\_notification\_id AND user\_id = sy-uname.       IF sy-subrc = 0.         ls\_zexx\_nf\_nrecip-rejected = abap\_true.         UPDATE zexx\_nf\_nrecip FROM ls\_zexx\_nf\_nrecip.       ENDIF.        send\_nf\_answer( EXPORTING iv\_nf\_id = iv\_notification\_id                                 iv\_action\_key = iv\_action\_key ).      WHEN 'Maybe'.       SELECT SINGLE \* FROM zexx\_nf\_nrecip INTO ls\_zexx\_nf\_nrecip WHERE notif\_id = iv\_notification\_id AND user\_id = sy-uname.       IF sy-subrc = 0.         ls\_zexx\_nf\_nrecip-maybe = abap\_true.         UPDATE zexx\_nf\_nrecip FROM ls\_zexx\_nf\_nrecip.       ENDIF.        send\_nf\_answer( EXPORTING iv\_nf\_id = iv\_notification\_id                                 iv\_action\_key = iv\_action\_key ).      WHEN OTHERS.   ENDCASE.    es\_result-success = abap\_true.  ENDMETHOD. |

* 1. Method /IWNGW/IF\_NOTIF\_PROVIDER~HANDLE\_BULK\_ACTION: This action will be performed when the user decides do accept all, reject all and so on. To clarify take a look at this picture:  
       
     

|  |
| --- |
| METHOD /iwngw/if\_notif\_provider~handle\_bulk\_action.    "Bulk actions are actions like: Approve all, reject all.   TYPES: BEGIN OF ty\_ids,            id TYPE zexx\_nf\_nrecip-notif\_id,          END OF ty\_ids.    DATA: lt\_ids TYPE STANDARD TABLE OF ty\_ids,         ls\_ids TYPE ty\_ids.    CHECK it\_bulk\_notif IS NOT INITIAL.    LOOP AT it\_bulk\_notif ASSIGNING FIELD-SYMBOL(<fs\_bulk\_notif>).     ls\_ids-id = <fs\_bulk\_notif>-id.     APPEND ls\_ids TO lt\_ids.   ENDLOOP.    SELECT \*     FROM zexx\_nf\_nrecip     INTO TABLE @DATA(lt\_zexx\_nf\_nrecip)     FOR ALL ENTRIES IN @lt\_ids     WHERE notif\_id = @lt\_ids-id       AND user\_id = @sy-uname.    CHECK sy-subrc = 0.    LOOP AT it\_bulk\_notif ASSIGNING <fs\_bulk\_notif>.     READ TABLE lt\_zexx\_nf\_nrecip ASSIGNING FIELD-SYMBOL(<fs\_zexx\_nf\_nrecip>) WITH KEY notif\_id = <fs\_bulk\_notif>-id.     IF sy-subrc = 0.       CASE <fs\_bulk\_notif>-action\_key.         WHEN 'Accept'.           <fs\_zexx\_nf\_nrecip>-accepted = abap\_true.            send\_nf\_answer( EXPORTING iv\_nf\_id = <fs\_bulk\_notif>-id                                     iv\_action\_key = <fs\_bulk\_notif>-action\_key ).          WHEN 'Reject'.           <fs\_zexx\_nf\_nrecip>-rejected = abap\_true.            send\_nf\_answer( EXPORTING iv\_nf\_id = <fs\_bulk\_notif>-id                                     iv\_action\_key = <fs\_bulk\_notif>-action\_key ).          WHEN 'Maybe'.           <fs\_zexx\_nf\_nrecip>-maybe = abap\_true.            send\_nf\_answer( EXPORTING iv\_nf\_id = <fs\_bulk\_notif>-id                                     iv\_action\_key = <fs\_bulk\_notif>-action\_key ).          WHEN OTHERS.       ENDCASE.     ENDIF.   ENDLOOP.    UPDATE zexx\_nf\_nrecip FROM TABLE lt\_zexx\_nf\_nrecip.  ENDMETHOD. |

* 1. Method /IWNGW/IF\_NOTIF\_PROVIDER~GET\_NOTIFICATION\_TYPE\_TEXT

|  |  |
| --- | --- |
| **Text** | **Table Source** |
| New message: &1 | &1: {message} |
| from: &1 | &1: {user} |

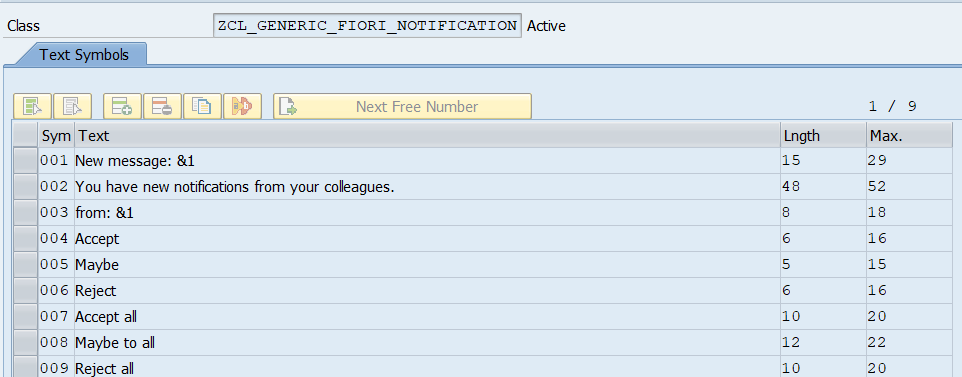
|  |
| --- |
| METHOD /iwngw/if\_notif\_provider~get\_notification\_type\_text.    DATA: lv\_text     TYPE string,         lv\_subtitle TYPE string.    DATA: ls\_action\_text TYPE LINE OF /iwngw/if\_notif\_provider=>ty\_t\_notification\_action\_text.    lv\_text = TEXT-001. "New message: &1    REPLACE '&1' WITH '{message}' INTO lv\_text.    lv\_subtitle = TEXT-003. "from: &1    REPLACE '&1' WITH '{user}' INTO lv\_subtitle.    es\_type\_text-subtitle            = lv\_subtitle.   es\_type\_text-template\_public     = lv\_text.   es\_type\_text-template\_grouped    = TEXT-002. "You have new notifications.   es\_type\_text-template\_sensitive  = lv\_text.    ls\_action\_text-action\_key   = 'Accept'.   ls\_action\_text-display\_text = TEXT-004. "Accept   ls\_action\_text-display\_text\_grouped = TEXT-007. "Accept all   APPEND ls\_action\_text TO et\_action\_text.    ls\_action\_text-action\_key   = 'Maybe'.   ls\_action\_text-display\_text = TEXT-005. "Maybe   ls\_action\_text-display\_text\_grouped = TEXT-008. "Maybe to all   APPEND ls\_action\_text TO et\_action\_text.    ls\_action\_text-action\_key   = 'Reject'.   ls\_action\_text-display\_text = TEXT-006. "Reject   ls\_action\_text-display\_text\_grouped = TEXT-009. "Reject all   APPEND ls\_action\_text TO et\_action\_text.  ENDMETHOD. |

* 1. Method /IWNGW/IF\_NOTIF\_PROVIDER~GET\_NOTIFICATION\_TYPE

|  |
| --- |
| METHOD /iwngw/if\_notif\_provider~get\_notification\_type.    DATA: ls\_action TYPE LINE OF /iwngw/if\_notif\_provider=>ty\_t\_notification\_action.    es\_notification\_type-is\_groupable = abap\_true.   es\_notification\_type-type\_key     = iv\_type\_key.   es\_notification\_type-version      = iv\_type\_version.    ls\_action-action\_key = 'Accept'.   ls\_action-nature = /iwngw/if\_notif\_provider=>gcs\_action\_natures-positive.   APPEND ls\_action TO et\_notification\_action.    ls\_action-action\_key = 'Maybe'.   ls\_action-nature = /iwngw/if\_notif\_provider=>gcs\_action\_natures-neutral.   APPEND ls\_action TO et\_notification\_action.    ls\_action-action\_key = 'Reject'.   ls\_action-nature = /iwngw/if\_notif\_provider=>gcs\_action\_natures-negative.   APPEND ls\_action TO et\_notification\_action.   ENDMETHOD. |

* 1. Method SEND\_NF\_ANSWER

|  |
| --- |
| METHOD send\_nf\_answer.    DATA: lt\_nf               TYPE /iwngw/if\_notif\_provider=>ty\_t\_notification,         ls\_nf               TYPE LINE OF /iwngw/if\_notif\_provider=>ty\_t\_notification,         lt\_user             TYPE STANDARD TABLE OF /iwngw/if\_notif\_provider=>ty\_s\_notification\_recipient,         ls\_user             TYPE /iwngw/if\_notif\_provider=>ty\_s\_notification\_recipient,         lt\_parameter\_values TYPE /iwngw/if\_notif\_provider=>ty\_t\_notification\_parameter,         ls\_parameter\_values TYPE LINE OF /iwngw/if\_notif\_provider=>ty\_t\_notification\_parameter,         ls\_parameter        TYPE LINE OF /iwngw/if\_notif\_provider=>ty\_t\_notification\_param\_bundle,         lt\_parameter        TYPE /iwngw/if\_notif\_provider=>ty\_t\_notification\_param\_bundle.    TRY.       DATA(lv\_system\_uuid) = cl\_uuid\_factory=>create\_system\_uuid( ).       ls\_nf-id             = lv\_system\_uuid->create\_uuid\_x16( ).     CATCH cx\_uuid\_error.       EXIT.   ENDTRY.    SELECT SINGLE \*          FROM zexx\_nf          INTO @DATA(ls\_zexx\_nf)          WHERE notif\_id = @iv\_nf\_id.   CHECK sy-subrc = 0.    ls\_user-id = ls\_zexx\_nf-created\_by.   APPEND ls\_user TO lt\_user. CLEAR ls\_user.    ls\_nf-type\_key                 = 'ZGENERIC\_NOTIFICATION'.   ls\_nf-type\_version             = '9'.   ls\_nf-recipients               = lt\_user[].    "Answer of the user: "X" to notification "Y": "Z".    ls\_parameter\_values-name         = 'notification'.   ls\_parameter\_values-type         = 'Edm.String'.   ls\_parameter\_values-value        = ls\_zexx\_nf-notification.   ls\_parameter\_values-is\_sensitive = abap\_false.   APPEND ls\_parameter\_values TO lt\_parameter\_values. CLEAR: ls\_parameter\_values.    ls\_parameter\_values-name         = 'user'.   ls\_parameter\_values-type         = 'Edm.String'.   ls\_parameter\_values-value        = sy-uname.   ls\_parameter\_values-is\_sensitive = abap\_false.   APPEND ls\_parameter\_values TO lt\_parameter\_values. CLEAR: ls\_parameter\_values.    ls\_parameter\_values-name         = 'decision'.   ls\_parameter\_values-type         = 'Edm.String'.   ls\_parameter\_values-value        = iv\_action\_key.   ls\_parameter\_values-is\_sensitive = abap\_false.   APPEND ls\_parameter\_values TO lt\_parameter\_values. CLEAR: ls\_parameter\_values.    ls\_parameter-language            = sy-langu.   ls\_parameter-parameters          = lt\_parameter\_values.   APPEND ls\_parameter TO lt\_parameter.    ls\_nf-parameters = lt\_parameter[].    APPEND ls\_nf TO lt\_nf.    TRY.       /iwngw/cl\_notification\_api=>create\_notifications(         EXPORTING           iv\_provider\_id  = 'ZGENERIC\_NF\_ANSWER' " class ZCL\_GENERIC\_NF\_ANSWER           it\_notification = lt\_nf ).        COMMIT WORK AND WAIT.      CATCH /iwngw/cx\_notification\_api INTO DATA(lrx\_api).       EXIT.   ENDTRY.  ENDMETHOD. |

* 1. Text elements  
     

1. Class ZCL\_GENERIC\_NF\_ANSWER
   1. Implement the interface /IWNGW/IF\_NOTIF\_PROVIDER
   2. Method /IWNGW/IF\_NOTIF\_PROVIDER~GET\_NOTIFICATION\_TYPE\_TEXT

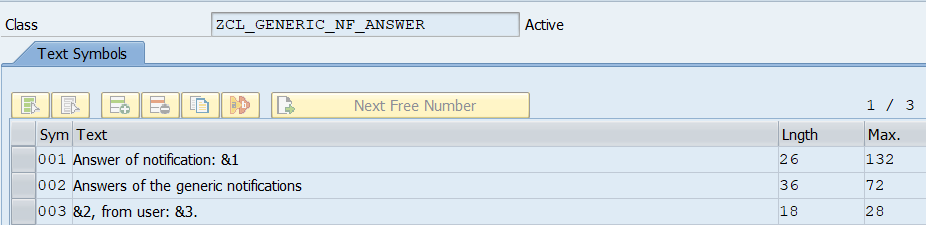
|  |  |
| --- | --- |
| **Text** | **Table Source** |
| Answer of notification: &1 | &1: {notification} |
| &2, from user: &3 | &2: {decision}  &3: {user} |

|  |
| --- |
| METHOD /iwngw/if\_notif\_provider~get\_notification\_type\_text.    DATA: lv\_text     TYPE string,         lv\_subtitle TYPE string.    lv\_text = TEXT-001. "Answer of notification: &1   lv\_subtitle = TEXT-003. "&2, from user: &3.    REPLACE '&1' WITH '{notification}' INTO lv\_text.   REPLACE '&2' WITH '{decision}' INTO lv\_subtitle.   REPLACE '&3' WITH '{user}' INTO lv\_subtitle.    es\_type\_text-template\_public     = lv\_text.   es\_type\_text-template\_grouped    = TEXT-002. "Answers of the generic notifications   es\_type\_text-template\_sensitive  = lv\_text.   es\_type\_text-subtitle  = lv\_subtitle.  ENDMETHOD. |

* 1. Method /IWNGW/IF\_NOTIF\_PROVIDER~GET\_NOTIFICATION\_TYPE

|  |
| --- |
| METHOD /IWNGW/IF\_NOTIF\_PROVIDER~GET\_NOTIFICATION\_TYPE.    es\_notification\_type-is\_groupable = abap\_true.   es\_notification\_type-type\_key     = iv\_type\_key.   es\_notification\_type-version      = iv\_type\_version.  ENDMETHOD. |

* 1. Text elements:



Conclusion

The SAP Fiori Notifications is a powerful tool to catch the attention of the users. In one place it is possible to configure the activities that should be performed in the system, categorized by priority, date or type.

The notification can even be a simple message or a important decision to be made, for example, an approval of purschase requisitions or sales orders.

It is well connected with standard workflows and can be very well customizable.

References

The content of this article was based in some SAP Notes, that were along the documentation described and the Fiori Guidelines.

Table of note references:

|  |  |
| --- | --- |
| [2488945](https://launchpad.support.sap.com/#/notes/2488945) | How to setup notifications in Fiori 2.0 |
| [2356153](https://launchpad.support.sap.com/#/notes/2356153) | Business Workflow Extensions - Version 2 |
| [2456330](https://launchpad.support.sap.com/#/notes/2456330) | How to configure Fiori Launchpad step-by-step |
| [2424054](https://launchpad.support.sap.com/#/notes/2424054) | How to setup My Inbox 2.0 app |
| [2729492](https://launchpad.support.sap.com/#/notes/2729492) | Configuring notifications in Fiori Launchpad and known restrictions |

Table of link references:

|  |
| --- |
| <https://wiki.scn.sap.com/wiki/display/SAPMOB/How+to+Setup+Notifications+in+Fiori+2.0+step-by-step> |
| <https://experience.sap.com/fiori-design-web/notification-center/> |