Class: ZCLPS_NT_EMAIL_ADDR_RETRIEVER

Status: Active

Attributes

Description: Email Addresses Retriever
Instantiation: Public
Not final

Not released
Fixed pt.arithmetic
Category: General Object Type
Package: ZBRPS
Original lang.: EN

Created by: ABAP
Created on: 03.03.2014
Last changed by: ABAP
Last changed on: 03.03.2014

Forward declaration type grps.

ABAP

Attribute

Public attributes

Attrib.	Cat	Description	Ref. Type	Init. value
AV_POSID	Inst (read only)	Work Breakdown Structure Element (WBS Element)	TYPE PS_POSID	
AV_ESTAT	Inst	User Status	TYPE J_ESTAT	

Protected attribute

Attrib.	Cat	Description	Ref. Type	Init. value
AT_EMADR	Inst	BCS: Internal Table with SMTP Addresses	TYPE BCSY_SMTPA	

Methods

Public methods

CONSTRUCTOR

Description: CONSTRUCTOR
Instance mthd

Importing parameter

`P_POSID TYPE PS_POSID` (Work Breakdown Structure Element (WBS Element))
`P_ESTAT TYPE J_ESTAT` (User Status)

```
method CONSTRUCTOR.
  me->av_posid = p_posid.
  me->av_estat = p_estat.
endmethod.
```

SET_EMAIL_ADDRESSES

Description: Fills in internal table AT_EMADR.
Instance mthd

```
METHOD set_email_addresses.
  DATA  lt_emadr     TYPE bcsy_smtpa.
  FIELD-SYMBOLS  <lv_email>  TYPE ad_smtpadr.
* Receivers for the event
  me->at_emadr = me->get_email_addr_from_zps0022( ).
* Users registered as receivers
  CLEAR lt_emadr.
  lt_emadr = me->get_email_addr_from_zps0021( ).
  IF lt_emadr IS NOT INITIAL.
    APPEND LINES OF lt_emadr TO me->at_emadr.
  ENDIF.
* WBS receivers (Exclusive)
  CLEAR lt_emadr.
  lt_emadr = me->get_email_addr_from_zps0004( ).
  IF lt_emadr IS NOT INITIAL.
    APPEND LINES OF lt_emadr TO me->at_emadr.
  ENDIF.
* Removes empty emails
  LOOP AT me->at_emadr ASSIGNING <lv_email>.
    IF <lv_email> IS INITIAL.
      DELETE me->at_emadr.
    ENDIF.
  ENDLOOP.
* Removes duplicates
  SORT me->at_emadr.
  DELETE ADJACENT DUPLICATES FROM me->at_emadr.
ENDMETHOD.
```

GET_EMAIL_ADDRESSES

Description: Returns the email addresses list.
Instance mthd

Returning parameter

`VALUE(RT_EMADR) TYPE BCSY_SMTPA` (BCS: Internal Table with SMTP Addresses)

```
method GET_EMAIL_ADDRESSES.
  IF me->at_emadr is INITIAL.
    me->set_email_addresses( ).
  ENDIF.
  rt_emadr = me->at_emadr.
endmethod.
```

ARE_THERE_RECIPIENTS

Description: Returns TRUE if there are recipients.
Instance mthd

Returning parameter

`VALUE(R_BOOLEAN) TYPE ABAP_BOOL` (Boolean variable.)

```
method ARE_THERE_RECIPIENTS.
```

```
  DATA lt_emadr TYPE BCSY_SMTPA.
  lt_emadr = me->get_email_addresses( ).
  IF lt_emadr is NOT INITIAL.
    r_boolean = abap_true.
  ENDIF.
endmethod.
```

GET_EMAIL_ADDR_FROM_ZPS0004

Description: Returns the email addresses list from ZPS0004.
Instance mthd

Returning parameter

`VALUE(RT_EMADR) TYPE BCSY_SMTPA` (BCS: Internal Table with SMTP Addresses)

```
method GET_EMAIL_ADDR_FROM_ZPS0004.
  DATA lo_db TYPE REF TO lcl_db.
  lo_db ?= me->get_db( ).
  CLEAR rt_emadr.
  TRY .
    rt_emadr = lo_db->get_wbs_emails( p_posid = me->av_posid
                                      p_estat = me->av_estat ).
  CATCH cx_sy_sql_error.
    CLEAR rt_emadr.
  ENDTRY.
  free lo_db.
ENDMETHOD.
```

GET_EMAIL_ADDR_FROM_ZPS0021

Description: Returns the email addresses list from ZPS0021.
Instance mthd

Returning parameter

`VALUE(RT_EMADR) TYPE BCSY_SMTPA` (BCS: Internal Table with SMTP Addresses)

```
method GET_EMAIL_ADDR_FROM_ZPS0021.
  DATA: lo_api       TYPE REF TO lcl_api,
        lt_emadr     TYPE bcsy_smtpa,
        lt_actors    TYPE tswhactor,
        lt_actors_ck TYPE tswhactor,
        lt_hr1001    TYPE hri1001_tab,
        lw_actor     TYPE swhactor,
        lv_uname     TYPE syuname,
        lv_sobid     TYPE sobid,
        lv_actid     TYPE ACTORID,
        lv_email     TYPE ad_smtpadr.
  FIELD-SYMBOLS:  <lw_actor>  TYPE swhactor,
                  <lw_hr1001> TYPE hri1001.
  lo_api ?= me->get_api( ).
  TRY .
* Users registered as receivers
    lt_actors = lo_api->rh_resolve_responsibilities(
    p_estat = me->av_estat ).
    LOOP AT lt_actors ASSIGNING <lw_actor>.
      IF <lw_actor>-otype = 'US'.
        lv_uname = <lw_actor>-objid.
        lv_email = lo_api->umb_service_user_email(
        p_uname = lv_uname ).
        APPEND lv_email TO rt_emadr.
        APPEND <lw_actor> TO lt_actors_ck.
        DELETE lt_actors.
      ELSE.
        CLEAR lt_hr1001.
        lv_sobid  = <lw_actor>-objid.
        TRY .
          lt_hr1001 = lo_api->rh_read_infty_1001_ext(
          p_otype = <lw_actor>-otype
          p_sobid = lv_sobid        ).
          DELETE: lt_hr1001 WHERE endda < sy-datum, " Not valid
                  lt_hr1001 WHERE rsign = 'A'
          AND relat = '002'. "Reports.
          LOOP AT lt_hr1001 ASSIGNING <lw_hr1001>.
            lv_actid = <lw_hr1001>-sobid.
            READ TABLE lt_actors_ck
              WITH KEY otype = <lw_hr1001>-sclas
                       objid = lv_actid
              TRANSPORTING NO FIELDS.
            IF sy-subrc <> 0.
              lw_actor-otype = <lw_hr1001>-sclas.
              lw_actor-objid = <lw_hr1001>-sobid.
              APPEND lw_actor TO lt_actors.
            ENDIF.
          ENDLOOP.
          APPEND <lw_actor> TO lt_actors_ck.
          DELETE lt_actors.
        CATCH cx_bapi_error.
          APPEND <lw_actor> TO lt_actors_ck.
          DELETE lt_actors.
          CONTINUE.
        ENDTRY.
      ENDIF.
    ENDLOOP.
  CATCH cx_bapi_error.
    CLEAR lt_emadr.
  ENDTRY.
  IF lt_emadr IS NOT INITIAL.
    APPEND LINES OF lt_emadr TO rt_emadr.
  ENDIF.
  free lo_api.
ENDMETHOD.
```

GET_EMAIL_ADDR_FROM_ZPS0022

Description: Returns the email addresses list from ZPS0022.
Instance mthd

Returning parameter

`VALUE(RT_EMADR) TYPE BCSY_SMTPA` (BCS: Internal Table with SMTP Addresses)

```
METHOD get_email_addr_from_zps0022.
  DATA  lo_db TYPE REF TO lcl_db.
  lo_db ?= me->get_db( ).
  TRY .
      rt_emadr = lo_db->get_event_emails( p_estat = me->av_estat ).
    CATCH cx_sy_sql_error.
      CLEAR rt_emadr.
  ENDTRY.
  FREE lo_db.
ENDMETHOD.
```

Protected methods

GET_API

Description: Returns a API calls mediator.
Instance mthd

Returning parameter

`VALUE(RO_API) TYPE REF TO OBJECT` (API calls mediator)

```
method GET_API.
  data lo_api TYPE REF TO lcl_api.
  create OBJECT lo_api.
  ro_api = lo_api.
  free lo_api.
endmethod.
```

GET_DB

Description: Returns a database query mediator.
Instance mthd

Returning parameter

`VALUE(RO_DB) TYPE REF TO OBJECT` (Database mediator.)

```
method GET_DB.
  DATA lo_db TYPE REF TO lcl_db.
  CREATE OBJECT lo_db.
  ro_db = lo_db.
  FREE lo_db.
endmethod.
```

Redefined Methods

Local Types

```
*"* use this source file for any type of declarations (class
*"* definitions, interfaces or type declarations) you need for
*"* components in the private section
```

Local class definitions

```
*"* use this source file for the definition and implementation of
*"* local helper classes, interface definitions and type
*"* declarations
*----------------------------------------------------------------------*
* Class LCL_API
*  OVERVIEW: Objects of this class mediates APIs calls. It aims to
*            isolate this development from external dependencies, as
*            might be desirable in tests.
*            The instances of this object is provided by the private
*            method GET_API of the main class
*            (ZCLPS_NT_EMAIL_ADDR_RETRIEVER). By redefining the method
*            GET_API one can provide appropriated stubs, which redefine
*            the desired methods of LCL_API, instead of calling the
*            real API.
*--------------------------------------------------------------------*
CLASS lcl_api DEFINITION.
  PUBLIC SECTION.
    METHODS: rh_resolve_responsibilities IMPORTING p_estat TYPE j_estat
                               RETURNING value(rt_actors) TYPE tswhactor
                               RAISING cx_bapi_error,
*--------------------------------------------------------------------*
* Method RH_RESOLVE_RESPONSIBILITIES
*--------------------------------------------------------------------*
* REQUIRES: The event code must be provided.
* EFFECTS.: If an error occurs during the call to FM
*           RH_RESOLVE_RESPONSIBILITIES raises an exeption of type
*           CX_BAPI_ERROR, otherwise the actors found
*--------------------------------------------------------------------*
* ->   P_ESTAT      Object status.
* <-   RT_ACTORS    Table with Organizational Objects.
*--------------------------------------------------------------------*
             umb_service_user_email IMPORTING p_uname TYPE syuname
                              RETURNING value(r_email) TYPE ad_smtpadr,
*--------------------------------------------------------------------*
* Method UMB_SERVICE_USER_EMAIL
*--------------------------------------------------------------------*
* REQUIRES: The user name must be provided.
* EFFECTS.: Returns the email address registered in the user master data
*--------------------------------------------------------------------*
* ->   P_UNAME      User Name.
* <-   R_EMAIL      E-Mail Address.
*--------------------------------------------------------------------*
             rh_read_infty_1001_ext IMPORTING p_otype TYPE otype
                                              p_sobid TYPE sobid
                          RETURNING value(rt_hri1001) TYPE hri1001_tab
                                    RAISING cx_bapi_error.
  PRIVATE SECTION.
    CONSTANTS: lcv_hr_object TYPE hrobjec_14 VALUE 'AC90300001',
               lcv_plvar     TYPE plvar      VALUE '01',
               lcv_element   TYPE swc_elem   VALUE 'STATUS'.
ENDCLASS.                    "lcl_api DEFINITION
*--------------------------------------------------------------------*
* Class LCL_DB
*  OVERVIEW: Objects of this class mediates database accesses. It aims
*            to isolate this development from external dependencies, as
*            might be desirable in tests.
*            The instances of this object is provided by the private
*            method GET_DB of the main class
*            (ZCLPS_NT_EMAIL_ADDR_RETRIEVER). By redefining the method
*            GET_DB one can provide appropriated stubs, which must
*            redefine the desired methods of LCL_DB, instead of
*            accessing real database tables.
*---------------------------------------------------------------------*
CLASS lcl_db DEFINITION.
  PUBLIC SECTION.
    METHODS: get_wbs_emails IMPORTING p_posid         TYPE ps_posid
                                      p_estat         TYPE j_estat
                            RETURNING value(rt_emadr) TYPE bcsy_smtpa
                            RAISING cx_sy_sql_error,
*---------------------------------------------------------------------*
* Method GET_WBS_EMAILS
*---------------------------------------------------------------------*
* REQUIRES: Key arguments for searching table ZTBPS_WF01 must be
*           provided.
* EFFECTS.: If no related data is found, returns exception type
*           CX_SX_SY_SQL_ERROR, otherwise returns the related emails
*           addresses list found in table ZTBPS_WF01.
*---------------------------------------------------------------------*
* ->   P_POSID      Work Breakdown Structure Element (WBS Element).
* ->   P_ESTAT      Object status.
* <-   RT_EMADR     Table with email addresses.
*---------------------------------------------------------------------*
             get_event_emails IMPORTING p_estat         TYPE j_estat
                              RETURNING value(rt_emadr) TYPE bcsy_smtpa
                              RAISING cx_sy_sql_error.
*---------------------------------------------------------------------*
* Method GET_EVENT_EMAILS
*---------------------------------------------------------------------*
* REQUIRES: The evend code must be provided.
* EFFECTS.: If no related data is found, returns exception type
*           CX_SX_SY_SQL_ERROR, otherwise returns the related emails
*           addresses list found in table ZTBPS_WF02.
*---------------------------------------------------------------------*
* ->   P_ESTAT      Object status.
* <-   RT_EMADR     Table with email addresses.
*---------------------------------------------------------------------*
ENDCLASS.                    "lcl_db DEFINITION
*----------------------------------------------------------------------*
*       CLASS lcl_api IMPLEMENTATION
*----------------------------------------------------------------------*
*
*----------------------------------------------------------------------*
CLASS lcl_api IMPLEMENTATION.
  METHOD rh_resolve_responsibilities.
    DATA: lt_container TYPE swconttab,
          lw_container TYPE swcont,
          lt_return    TYPE bapiret2_t,
          lw_return    TYPE bapiret2.
    lw_container-element = lcv_element.
    lw_container-value   = p_estat.
    APPEND lw_container TO lt_container.
    CALL FUNCTION 'RH_RESOLVE_RESPONSIBILITIES'
      EXPORTING
        act_object      = lcv_hr_object
        act_plvar       = lcv_plvar
      TABLES
        ac_container    = lt_container
        actor_tab       = rt_actors
      EXCEPTIONS
        no_active_plvar = 1
        no_actor_found  = 2
        parameter_error = 3
        OTHERS          = 4.
    IF sy-subrc <> 0.
      m_raise_bapi_exeption sy-msgty sy-msgid sy-msgno
                            sy-msgv1 sy-msgv2 sy-msgv3
                            sy-msgv4.
    ENDIF.
  ENDMETHOD.                    "RH_RESOLVE_RESPONSIBILITIES
  METHOD umb_service_user_email.
    CALL FUNCTION 'UMB_SERVICE_USER_EMAIL'
      EXPORTING
        i_user  = p_uname
      IMPORTING
        e_email = r_email.
  ENDMETHOD.                    "UMB_SERVICE_USER_EMAIL
  METHOD rh_read_infty_1001_ext.
    DATA: lt_objects   TYPE hrtb_sobid,
          lw_object    TYPE hrsobid,
          lt_return    TYPE bapiret2_t,
          lw_return    TYPE bapiret2.
    lw_object-plvar = lcv_plvar.
    lw_object-otype = p_otype.
    lw_object-sobid = p_sobid.
    APPEND lw_object TO lt_objects.
    CALL FUNCTION 'RH_READ_INFTY_1001_EXT'
      TABLES
        i1001           = rt_hri1001
        objects         = lt_objects
      EXCEPTIONS
        nothing_found   = 1
        wrong_condition = 2
        OTHERS          = 3.
    IF sy-subrc <> 0.
      m_raise_bapi_exeption sy-msgty sy-msgid sy-msgno
                            sy-msgv1 sy-msgv2 sy-msgv3
                            sy-msgv4.
    ENDIF.
  ENDMETHOD.                    "rh_read_infty_1001_ext
ENDCLASS.                    "lcl_api IMPLEMENTATION
*----------------------------------------------------------------------*
*       CLASS lcl_db IMPLEMENTATION
*----------------------------------------------------------------------*
*
*----------------------------------------------------------------------*
CLASS lcl_db IMPLEMENTATION.
  METHOD get_wbs_emails.
    SELECT smtp_addr FROM ztbps_wf01
      INTO TABLE rt_emadr
```

```
     WHERE posid = p_posid
       AND estat = p_estat
       AND ativo = 'X'.
    IF sy-subrc <> 0.
      RAISE EXCEPTION TYPE cx_sy_sql_error.
    ENDIF.
  ENDMETHOD.                    "get_wbs_emails
  METHOD get_event_emails.
    SELECT smtp_addr FROM ztbps_wf02
      INTO TABLE rt_emadr
     WHERE estat = p_estat
       AND activ = 'X'.
    IF sy-subrc <> 0.
      RAISE EXCEPTION TYPE cx_sy_sql_error.
    ENDIF.
  ENDMETHOD.                    "get_event_emails
ENDCLASS.                    "lcl_db IMPLEMENTATION
```

Macros

```
*"* use this source file for any macro definitions you need
*"* in the implementation part of the class
DEFINE m_raise_bapi_exeption.
  lw_return-TYPE       = &1.
  lw_return-ID         = &2.
  lw_return-NUMBER     = &3.
  lw_return-message_v1 = &4.
  lw_return-message_v2 = &5.
  lw_return-message_v3 = &6.
  lw_return-message_v4 = &7.
  MESSAGE ID &2 TYPE &1 NUMBER &3 WITH &4 &5 &6 &7
     INTO lw_return-MESSAGE.
  APPEND lw_return TO lt_return.
  RAISE EXCEPTION TYPE cx_bapi_error
  EXPORTING
    status = lt_return.
END-OF-DEFINITION.
```

Overview