Class: ZCL_TVARV_RETRIEVER

Status: Active

Attributes

Description: TVARV Retriever
Instantiation: Public
Not final

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

Created by: ABAP
Created on: 11.03.2014
Last changed by: ABAP
Last changed on: 11.03.2014

Attribute

Public attributes

Attrib.	Cat	Description	Ref. Type	Init. value
AV_NAME	Inst (read only)	ABAP: Name of Variant Variable	TYPE RVARI_VNAM	

Protected attribute

Attrib.	Cat	Description	Ref. Type	Init. value
AT_VALUES	Inst	Table for TVARVC Structure	TYPE TVARVC_T	

Methods

Public methods

CONSTRUCTOR
Description: CONSTRUCTOR
Instance mthd

Importing parameter
  P_NAME TYPE RVARI_VNAM (ABAP: Name of Variant Variable)

```
method CONSTRUCTOR.
  me->av_name = p_name.
endmethod.
```

GET_SINGLE
Description: Returns a single value of this object.
Instance mthd

Importing parameter
  I_VNAME TYPE STRING OPTIONAL (Name of a variable.)
Exporting parameter
  VALUE(EX_VALUE) TYPE ANY (Value.)

```
METHOD get_single.
  DATA  lv_search TYPE rvari_vnam.
  FIELD-SYMBOLS <lw_value> TYPE tvarvc.
  IF me->at_values IS INITIAL.
    me->set_attributes( ).
  ENDIF.
  IF i_vname IS SUPPLIED.
    CONCATENATE av_name i_vname INTO lv_search SEPARATED BY '_'.
    READ TABLE me->at_values WITH KEY name = lv_search
     ASSIGNING <lw_value>.
    IF sy-subrc = 0.
      ex_value = <lw_value>-low.
    ENDIF.
  ELSE.
    READ TABLE me->at_values INDEX 1 ASSIGNING <lw_value>.
    IF sy-subrc = 0.
      ex_value = <lw_value>-low.
    ENDIF.
  ENDIF.
ENDMETHOD.
```

GET_MULTIPLE
Description: Returns the range of values of this object.
Instance mthd

Importing parameter
```
  I_VNAME TYPE STRING  OPTIONAL (Name of a variable.)
```

Exporting parameter
```
  VALUE(EXT_VALUES) TYPE ANY TABLE (Table with the range of values of this object.)
```

```
METHOD get_multiple.
  TYPES  lyr_tvarv TYPE RANGE OF tvarv_val.
  DATA: lv_search TYPE rvari_vnam,
        lr_tvarv  TYPE lyr_tvarv,
        lw_tvarv  TYPE LINE OF lyr_tvarv.
  FIELD-SYMBOLS <lw_value> TYPE tvarvc.
  DEFINE lm_append.
    lw_tvarv-sign   = <lw_value>-sign.
    lw_tvarv-option = <lw_value>-opti.
    lw_tvarv-low    = <lw_value>-low.
    lw_tvarv-high   = <lw_value>-high.
    append lw_tvarv to lr_tvarv.
  END-OF-DEFINITION.
  IF me->at_values IS INITIAL.
    me->set_attributes( ).
  ENDIF.
  IF i_vname IS SUPPLIED.
    CONCATENATE av_name i_vname INTO lv_search SEPARATED BY '_'.
    LOOP AT me->at_values ASSIGNING <lw_value> WHERE name = lv_search.
      lm_append.
    ENDLOOP.
  ELSE.
    LOOP AT me->at_values ASSIGNING <lw_value>.
      lm_append.
```

```
    ENDLOOP.
  ENDIF.
  ext_values = lr_tvarv.
ENDMETHOD.
```

Protected methods

SET_ATTRIBUTES
Description: Retrieves the attributes of this object.
Instance mthd

```
method SET_ATTRIBUTES.
  DATA lo_db TYPE REF TO lcl_db.
  lo_db ?= me->get_db( ).
  TRY .
    lo_db->get_tvarvc_values( EXPORTING
                                 p_vname = me->av_name
                              IMPORTING
                                 rt_values = me->AT_VALUES ).
  CATCH cx_sy_sql_error.
    free lo_db. return.
  ENDTRY.
  sort me->at_values.
  free lo_db.
endmethod.
```

GET_DB
Description: Returns an database 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_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 (ZCL_TVARV_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_tvarvc_values IMPORTING p_vname          TYPE rvari_vnam
                               EXPORTING value(rt_values) TYPE TVARVC_T
                                RAISING cx_sy_sql_error.
*---------------------------------------------------------------------*
* Method GET_TVARVC_VALUES
*---------------------------------------------------------------------*
* REQUIRES: The variant name must be provided.
* EFFECTS.: If no related data is found, returns exception type
*           CX_SX_SY_SQL_ERROR, otherwise returns the related variant
*           type and a table with the range of related values found in
*           tabel TVARVC.
*---------------------------------------------------------------------*
* ->   P_VNAME     Name of Variant Variable.
* <-   RT_VALUES   Collection of variant values.
*---------------------------------------------------------------------*
ENDCLASS.                    "lcl_db DEFINITION
*---------------------------------------------------------------------*
*      CLASS lcl_db IMPLEMENTATION
*---------------------------------------------------------------------*
*
*---------------------------------------------------------------------*
CLASS lcl_db IMPLEMENTATION.
  METHOD get_tvarvc_values.
    SELECT * FROM tvarvc
      INTO TABLE rt_values
     WHERE name LIKE p_vname.
    IF sy-subrc <> 0.
      RAISE EXCEPTION TYPE cx_sy_sql_error.
    ENDIF.
  ENDMETHOD.                    "get_tvarvc_values
ENDCLASS.                    "lcl_db IMPLEMENTATION
```

Macros

```
*"* use this source file for any macro definitions you need
*"* in the implementation part of the class
```

Overview