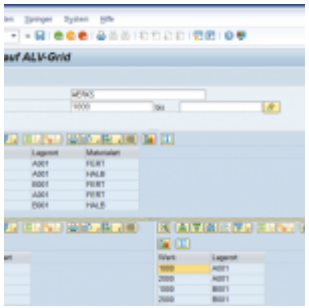


Magic Filter

Vor kurzem war ich genervt von einer Anwendung, in der Daten in mehreren ALV-Grids angezeigt wurden. Die unterschiedlichen Grids haben teilweise gleiche Felder. Vielleicht ähnlich eines Cockpits in dem verschiedene Sichten zu Materialien angezeigt werden (Materialstamm, Werks-Sichten, Bestände auf Lagerortebene etc.)

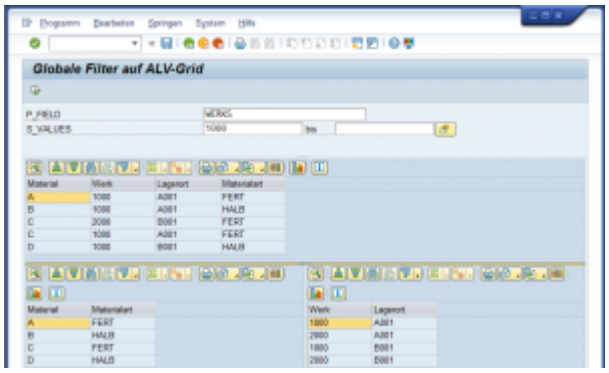
Bei der Analyse bzw. Fehlersuche musste ich in mehreren Grids einen Filter setzen, z.B. auf das Werk. Das bei verschiedenen Grids ist zwar recht schnell gemacht, aber als ich es öfters machen musste, nervte es schon ziemlich.

Magic Filter

Dabei hatte ich die Idee, einen Filter global über alle verwendeten Grids zu setzen. Die Idee hat sich in dem unten zur Verfügung gestellten Programm manifestiert. Die Funktionalität ist in der Klasse gekapselt. Der Rest des Codes dient nur dazu, einen dreigeteilten Splitter mit jeweils einem Grid darstellen zu können.

Jedes Grid registriert sich am Magic-Filter-Controller. In P_FIELD kann ein Feldname gesetzt werden – z.B. WERKS – und in S_VALUES können die Werte eingeschränkt werden. Mit <ENTER> wird der Filter auf alle registrierten Grids angewendet.

Sofern ein Feld nicht im Feldkatalog des Grids ist, wird es vor dem Setzen des Filters gelöscht.



Code

```
REPORT zz_magic_filter.

class lcl_magic_filter DEFINITION DEFERRED.
DATA gr_gfil      TYPE REF TO lcl_magic_filter.
```

```
DATA gv_value     TYPE c LENGTH 20.

DATA gs_filter    TYPE lvc_s_filt.
DATA gt_filter    TYPE lvc_t_filt.

DATA gr_docker    TYPE REF TO cl_gui_docking_container.

DATA gr_splitter1 TYPE REF TO cl_gui_easy_splitter_container.
DATA gr_splitter2 TYPE REF TO cl_gui_easy_splitter_container.

DATA gr_cont1     TYPE REF TO cl_gui_container.
DATA gr_cont2     TYPE REF TO cl_gui_container.
DATA gr_cont3     TYPE REF TO cl_gui_container.
DATA gr_cont4     TYPE REF TO cl_gui_container.

DATA gr_grid1     TYPE REF TO cl_gui_alv_grid.
DATA gr_grid2     TYPE REF TO cl_gui_alv_grid.
DATA gr_grid3     TYPE REF TO cl_gui_alv_grid.

TYPES: BEGIN OF ty_1,
         matnr TYPE matnr,
         werks TYPE werks_d,
         lgort TYPE lgort_d,
         mtart TYPE mtart,
       END OF ty_1.

TYPES: BEGIN OF ty_2,
         matnr TYPE matnr,
         mtart TYPE mtart,
       END OF ty_2.

TYPES: BEGIN OF ty_3,
         werks TYPE werks_d,
         lgort TYPE lgort_d,
       END OF ty_3.

DATA gt_1 TYPE STANDARD TABLE OF ty_1.
DATA gt_2 TYPE STANDARD TABLE OF ty_2.
DATA gt_3 TYPE STANDARD TABLE OF ty_3.

DATA gs_1 TYPE ty_1.
DATA gs_2 TYPE ty_2.
DATA gs_3 TYPE ty_3.


CLASS lcl_magic_filter DEFINITION.

  PUBLIC SECTION.

    METHODS register
      IMPORTING
```

```
        ir_grid TYPE REF TO cl_gui_alv_grid .
    METHODS set_filter
      IMPORTING
        filter TYPE lvc_t_filt .
  PROTECTED SECTION.

    TYPES:
      BEGIN OF ty_object,
        grid   TYPE REF TO cl_gui_alv_grid,
        fcat   TYPE lvc_t_fcat,
        filter TYPE lvc_t_filt,
        status TYPE c LENGTH 1,
      END OF ty_object .
    TYPES:
      ty_objects TYPE STANDARD TABLE OF ty_object .

    DATA mt_objects TYPE ty_objects .
    DATA mt_filter TYPE lvc_t_filt .

    METHODS set_filter_on_objects .

ENDCLASS.



CLASS lcl_magic_filter IMPLEMENTATION.


* ---------------------------------------------------------------------
------------+
* | Instance Public Method ZCL_GFIL_CONTROLLER->REGISTER
* +---------------------------------------------------------------------
-----------------------+
* | [--->] IR_GRID                        TYPE REF TO CL_GUI_ALV_GRID
* +---------------------------------------------------------------------
------------
  METHOD register.

    DATA ls_object LIKE LINE OF mt_objects.

    READ TABLE mt_objects TRANSPORTING NO FIELDS WITH KEY grid = ir_grid.
    IF sy-subrc > 0.
*== Objekt hinzufügen
      ls_object-grid   = ir_grid.
      ls_object-status = '1'.
      ir_grid->get_frontend_fieldcatalog( IMPORTING et_fieldcatalog =
ls_object-fcat ).
      APPEND ls_object TO mt_objects.
    ENDIF.

  ENDMETHOD.
```

```
* ---------------------------------------------------------------------
------------+
* | Instance Public Method ZCL_GFIL_CONTROLLER->SET_FILTER
* +---------------------------------------------------------------------
----------------------+
* | [--->] FILTER                         TYPE        LVC_T_FILT
* +---------------------------------------------------------------------
------------
  METHOD set_filter.

    mt_filter = filter.
    set_filter_on_objects( ).

  ENDMETHOD.


* ---------------------------------------------------------------------
------------+
* | Instance Protected Method ZCL_GFIL_CONTROLLER->SET_FILTER_ON_OBJECTS
* +---------------------------------------------------------------------
----------------------+
* +---------------------------------------------------------------------
------------
  METHOD set_filter_on_objects.

    DATA lt_filter TYPE lvc_t_filt.
    DATA lv_index TYPE i.

*== Set filter
    LOOP AT mt_objects ASSIGNING FIELD-SYMBOL(<object>).
      lt_filter = mt_filter.

      LOOP AT lt_filter ASSIGNING FIELD-SYMBOL(<filter>).
        lv_index = sy-tabix.
        READ TABLE <object>-fcat TRANSPORTING NO FIELDS WITH KEY fieldname =
<filter>-fieldname.
        IF sy-subrc > 0.
          DELETE lt_filter INDEX lv_index.
        ENDIF.
        <object>-grid->set_filter_criteria( lt_filter ).
      ENDLOOP.
      <object>-grid->refresh_table_display( is_stable = VALUE #( col = 'X'
row = 'X' )
                                            i_soft_refresh = space ).
    ENDLOOP.


  ENDMETHOD.
ENDCLASS.
```

```
PARAMETER p_field TYPE fieldname.
SELECT-OPTIONS s_values FOR gv_value.

INITIALIZATION.

  CREATE OBJECT gr_gfil.

  PERFORM create_base.
  PERFORM create_1.
  PERFORM create_2.
  PERFORM create_3.



AT SELECTION-SCREEN.

  CLEAR gs_filter.
  CLEAR gt_filter.
  gs_filter-fieldname = p_field.
  LOOP AT s_values.
    gs_filter-low    = s_values-low.
    gs_filter-sign   = s_values-sign.
    gs_filter-option = s_values-option.
    APPEND gs_filter TO gt_filter.
  ENDLOOP.

  gr_gfil->set_filter( gt_filter ).

START-OF-SELECTION.

FORM create_base.


  CREATE OBJECT gr_docker
    EXPORTING
      side                    = cl_gui_docking_container=>dock_at_bottom
      ratio                   = 80
      no_autodef_progid_dynnr = abap_true.


  CREATE OBJECT gr_splitter1
    EXPORTING
      parent        = gr_docker
      orientation   = 0
      sash_position = 50.

  gr_cont1 = gr_splitter1->top_left_container.
  gr_cont4 = gr_splitter1->bottom_right_container.

  CREATE OBJECT gr_splitter2
    EXPORTING
```

```
      parent        = gr_cont4
      orientation   = 1
      sash_position = 50.

  gr_cont2 = gr_splitter2->top_left_container.
  gr_cont3 = gr_splitter2->bottom_right_container.

ENDFORM.

FORM create_1.

  DATA ls_fcat TYPE lvc_s_fcat.
  DATA lt_fcat TYPE lvc_t_fcat.

  gs_1-matnr = 'A'.
  gs_1-werks = '1000'.
  gs_1-lgort = 'A001'.
  gs_1-mtart = 'FERT'.
  APPEND gs_1 TO gt_1.

  gs_1-matnr = 'B'.
  gs_1-werks = '1000'.
  gs_1-lgort = 'A001'.
  gs_1-mtart = 'HALB'.
  APPEND gs_1 TO gt_1.

  gs_1-matnr = 'C'.
  gs_1-werks = '2000'.
  gs_1-lgort = 'B001'.
  gs_1-mtart = 'FERT'.
  APPEND gs_1 TO gt_1.

  gs_1-matnr = 'C'.
  gs_1-werks = '1000'.
  gs_1-lgort = 'A001'.
  gs_1-mtart = 'FERT'.
  APPEND gs_1 TO gt_1.

  gs_1-matnr = 'D'.
  gs_1-werks = '1000'.
  gs_1-lgort = 'B001'.
  gs_1-mtart = 'HALB'.
  APPEND gs_1 TO gt_1.

  ls_fcat-fieldname = 'MATNR'.
  ls_fcat-rollname  = 'MATNR'.
  APPEND ls_fcat TO lt_fcat.
  ls_fcat-fieldname = 'WERKS'.
  ls_fcat-rollname  = 'WERKS_D'.
  APPEND ls_fcat TO lt_fcat.
  ls_fcat-fieldname = 'LGORT'.
```

```
  ls_fcat-rollname  = 'LGORT_D'.
  APPEND ls_fcat TO lt_fcat.
  ls_fcat-fieldname = 'MTART'.
  ls_fcat-rollname  = 'MTART'.
  APPEND ls_fcat TO lt_fcat.

  PERFORM create_grid USING gr_cont1 lt_fcat gt_1.

ENDFORM.

FORM create_2.

  DATA ls_fcat TYPE lvc_s_fcat.
  DATA lt_fcat TYPE lvc_t_fcat.

  gs_2-matnr = 'A'.
  gs_2-mtart = 'FERT'.
  APPEND gs_2 TO gt_2.
  gs_2-matnr = 'B'.
  gs_2-mtart = 'HALB'.
  APPEND gs_2 TO gt_2.
  gs_2-matnr = 'C'.
  gs_2-mtart = 'FERT'.
  APPEND gs_2 TO gt_2.
  gs_2-matnr = 'D'.
  gs_2-mtart = 'HALB'.
  APPEND gs_2 TO gt_2.


  ls_fcat-fieldname = 'MATNR'.
  ls_fcat-rollname  = 'MATNR'.
  APPEND ls_fcat TO lt_fcat.
  ls_fcat-fieldname = 'MTART'.
  ls_fcat-rollname  = 'MTART'.
  APPEND ls_fcat TO lt_fcat.

  PERFORM create_grid USING gr_cont2 lt_fcat gt_2.

ENDFORM.

FORM create_3.

  DATA ls_fcat TYPE lvc_s_fcat.
  DATA lt_fcat TYPE lvc_t_fcat.


  gs_3-werks = '1000'.
  gs_3-lgort = 'A001'.
  APPEND gs_3 TO gt_3.
  gs_3-werks = '2000'.
  gs_3-lgort = 'A001'.
```

```
  APPEND gs_3 TO gt_3.
  gs_3-werks = '1000'.
  gs_3-lgort = 'B001'.
  APPEND gs_3 TO gt_3.
  gs_3-werks = '2000'.
  gs_3-lgort = 'B001'.
  APPEND gs_3 TO gt_3.



  ls_fcat-fieldname = 'WERKS'.
  ls_fcat-rollname  = 'WERKS_D'.
  APPEND ls_fcat TO lt_fcat.
  ls_fcat-fieldname = 'LGORT'.
  ls_fcat-rollname  = 'LGORT_D'.
  APPEND ls_fcat TO lt_fcat.

  PERFORM create_grid USING gr_cont3 lt_fcat gt_3.

ENDFORM.

FORM create_grid USING ir_container TYPE REF TO cl_gui_container
                       it_fcat      TYPE lvc_t_fcat
                       it_table     TYPE table.

  DATA lr_grid TYPE REF TO cl_gui_alv_grid.

  CREATE OBJECT lr_grid
    EXPORTING
      i_parent = ir_container.

  lr_grid->set_table_for_first_display(
    CHANGING
      it_outtab                     = it_table
      it_fieldcatalog               = it_fcat ).

  gr_gfil->register( lr_grid ).

ENDFORM.
```