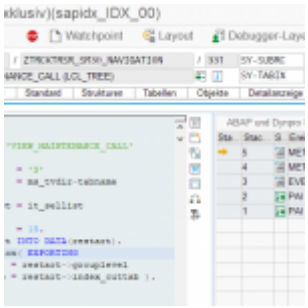

Call Stack umgehen

In dem Artikel Pflegeview mit Datennavigation habe ich eine Möglichkeit vorgestellt, wie man Daten mit Hilfe einer Treedarstellung besser visualisieren und bearbeiten kann.

Leider gab es hier Umstand, dass mit jedem Doppelklick auf einen Eintrag im Tree ein neuer Pflegedialog aufgerufen wurde (Call Stack). Mit jeder Navigation wird also ein CALL SCREEN gemacht und somit der Call Stack erhöht. Der Call Stack ist auf eine bestimmte Anzahl Aufrufe beschränkt (ca. 60). Selbst wenn der Call Stack höher wäre und somit „anwendertauglich“, so wäre es doch irgendwie *falsch*, es so zu machen.

Ein Freund hat dann den entscheidenden Hinweis gefunden, um den Call Stack zu durchbrechen.

Exception Class

Wir benötigen dazu eine eigene Ausnahmeklasse, die von CX_NO_CHECK erbt und die Parameter GROUPLEVEL und INDEX_OUTTAB als Parameter hat:

```
CLASS lcx_restart DEFINITION INHERITING FROM cx_no_check.
  PUBLIC SECTION.
    DATA grouplevel TYPE lvc_fname.
    DATA index_outtab TYPE lvc_index.
    METHODS constructor
      IMPORTING
        grouplevel   TYPE lvc_fname
        index_outtab TYPE lvc_index.
ENDCLASS.

CLASS lcx_restart IMPLEMENTATION.
  METHOD  constructor.
    CALL METHOD super->constructor
      EXPORTING
        textid   = textid
        previous = previous.
    me->index_outtab = index_outtab .
    me->grouplevel = grouplevel .
  ENDMETHOD.
ENDCLASS.
```

Hilfsklasse

Wir benötigen eine Hilfsklasse, die die Ausnahme aufruft und den VIEW_MAINTENANCE_CALL beendet:

```
CLASS lcl_helper IMPLEMENTATION.
  METHOD handle_item_double_click.
    handle_node_double_click(
      grouplevel   = grouplevel
      index_outtab = index_outtab ).
  ENDMETHOD.

  METHOD handle_node_double_click.

    RAISE EXCEPTION TYPE lcx_restart
      EXPORTING
        grouplevel   = grouplevel
        index_outtab = index_outtab.
  ENDMETHOD.

  METHOD install_handler.
    tree = i_tree.
    SET HANDLER handle_node_double_click FOR tree.
    SET HANDLER handle_item_double_click FOR tree.
  ENDMETHOD.
ENDCLASS.
```

Ereignisregistrierung

Die Ereignisregistrierung kann weiterhin in der *Navigationsklasse* erfolgen. Die Handler müssen jedoch in der externen Klasse LCL_HELPER installiert werden:

```
 METHOD register_events.

    lcl_helper=>install_handler( mo_tree ).

    mo_tree->set_registered_events( VALUE #(
          "Used here for applying current data selection
          ( eventid = cl_gui_column_tree=>eventid_node_double_click )
          ( eventid = cl_gui_column_tree=>eventid_item_double_click )
          "Important! If not registered nodes will not expand ->No data
          ( eventid = cl_gui_column_tree=>eventid_expand_no_children ) ) ) .

  ENDMETHOD.
```

Zur Installation der Handler übergeben wir einfach die Instanz des Trees um in der Hilfsklasse auf die Ereignisse reagieren zu können.

View_Maintenance_Call

Der eigentliche Clou ist jedoch, dass wir den Aufruf des Funktionsbaustein VIEW_MAINTENANCE_CALL mit einem TRY-CATCH-Block kapseln:

```
  METHOD view_maintenance_call.
    TRY.
        CALL FUNCTION 'VIEW_MAINTENANCE_CALL'
          EXPORTING
            action      = 'S'
            view_name   = ms_tvdir-tabname
          TABLES
            dba_sellist = it_sellist
          EXCEPTIONS
            OTHERS      = 15.
      CATCH lcx_restart INTO DATA(restart).
        handle_selection( EXPORTING
          grouplevel   = restart->grouplevel
          index_outtab = restart->index_outtab ).
    ENDTRY.
  ENDMETHOD.
```

Wir das Ereignis LCX_RESTART ausgelöst, dann starten wir den Pflegedialog einfach erneut. Allerdings nun mit einem abgebauten Call Stack.

Call Stack umgehen

Den Call Stack durch einen externen RAISE EXCEPTION abzubrechen könnte auch in anderen Fällen hilfreich sein. In welchen, das musst du selber herausfinden…