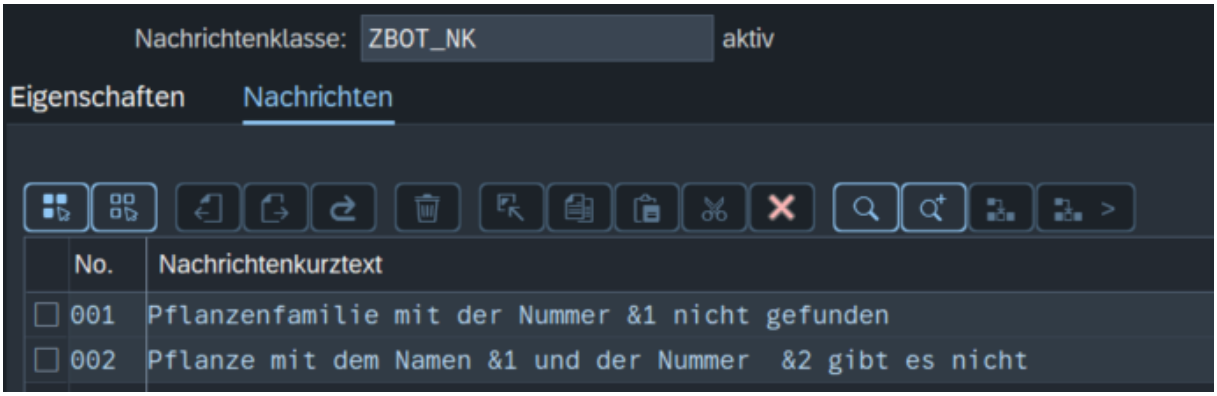
Praktikum 12 Lösung

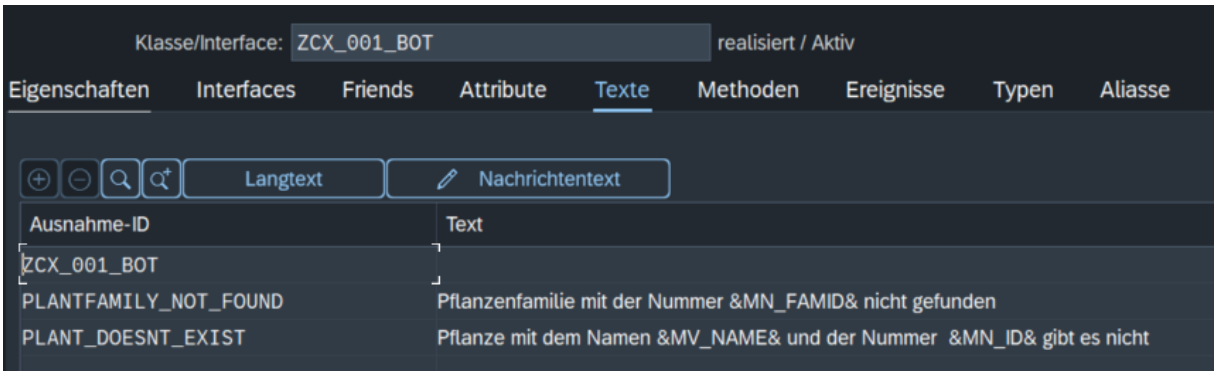
1.Nachrichtenklasse



2.Ausnahmeklasse

Attribute

Texte



3. Objektklasse PLANT

```
CLASS zbot_obj DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC .

  PUBLIC SECTION.

    EVENTS plant_created
      EXPORTING
        VALUE(eo_plant) TYPE REF TO zbot_obj .

    METHODS constructor
      IMPORTING
        VALUE(is_plant) TYPE zbot_testdaten
      RAISING
        zcx_001_bot .
    METHODS get_data
      RETURNING
        VALUE(rs_plant) TYPE zbot_testdaten .
  PROTECTED SECTION.
  PRIVATE SECTION.

    DATA ms_plant TYPE zbot_testdaten.
ENDCLASS.



CLASS ZBOT_OBJ IMPLEMENTATION.


* <SIGNATURE>---------------------------------------------------------------------------------------+
* | Instance Public Method ZBOT_OBJ->CONSTRUCTOR
* +--------------------------------------------------------------------------------------------------+
* | [--->] IS_PLANT                       TYPE        TS_PLANT
* | [!CX!] ZCX_001_BOT
* +---------------------------------------------------------------------------------------</SIGNATURE>
  METHOD constructor.
    SELECT SINGLE * FROM zbot_family
      INTO @DATA(ls_family)
      WHERE famid = @is_plant-famid.
    IF sy-subrc <> 0.
      RAISE EXCEPTION TYPE zcx_001_bot
        EXPORTING
          textid   = zcx_001_bot=>plantfamily_not_found
          mn_famid = is_plant-famid.
    ENDIF.

    SELECT SINGLE * FROM zbot_plant
    INTO @DATA(ls_plant)
    WHERE plantid = @is_plant-plantid.
    IF sy-subrc <> 0 OR ( ls_plant-name <> is_plant-name ).
      RAISE EXCEPTION TYPE zcx_001_bot
        EXPORTING
          textid  = zcx_001_bot=>plant_doesnt_exist
          mn_id   = is_plant-plantid
          mv_name = is_plant-name.
    ENDIF.
    ms_plant = is_plant.
    RAISE EVENT plant_created EXPORTING eo_plant = me.
  ENDMETHOD.


* <SIGNATURE>---------------------------------------------------------------------------------------+
* | Instance Public Method ZBOT_OBJ->GET_DATA
* +--------------------------------------------------------------------------------------------------+
* | [<-()] RS_PLANT                       TYPE        TS_PLANT
* +---------------------------------------------------------------------------------------</SIGNATURE>
  METHOD get_data.
    rs_plant = ms_plant.
  ENDMETHOD.
ENDCLASS.
```

4. Poolklasse

```
class ZBOT_POOL definition
  public
  final
  create public .

public section.

  methods ON_PLANT_CREATED
    for event PLANT_CREATED of ZBOT_OBJ
    importing
      !EO_PLANT .
  methods ANZAHL
    returning
      value(RN_ANZAHL) type I .
  methods ANZ_PLANT
    importing
      value(IN_PLANTID) type ZBOT_PLANTID
    returning
      value(RN_ANZAHL) type I .
  methods AUSGABE .
  PROTECTED SECTION.
private section.

  data:
    mt_o_plant TYPE TABLE OF REF TO zbot_obj .

  methods ADD_PLANT
    importing
      !IO_PLANT type ref to ZBOT_OBJ .
ENDCLASS.



CLASS ZBOT_POOL IMPLEMENTATION.


* <SIGNATURE>----------------------------------------------------------------------------------------+
* | Instance Private Method ZBOT_POOL->ADD_PLANT
* +--------------------------------------------------------------------------------------------------+
* | [--->] IO_PLANT                       TYPE REF TO ZBOT_OBJ
* +--------------------------------------------------------------------------------------</SIGNATURE>
  METHOD add_plant.
    APPEND io_plant TO mt_o_plant.
  ENDMETHOD.


* <SIGNATURE>----------------------------------------------------------------------------------------+
* | Instance Public Method ZBOT_POOL->ANZAHL
* +--------------------------------------------------------------------------------------------------+
* | [<-()] RN_ANZAHL                      TYPE        I
* +--------------------------------------------------------------------------------------</SIGNATURE>
  method ANZAHL.
    rn_anzahl = lines( mt_o_plant ).
  endmethod.


* <SIGNATURE>----------------------------------------------------------------------------------------+
* | Instance Public Method ZBOT_POOL->ANZ_PLANT
* +--------------------------------------------------------------------------------------------------+
* | [--->] IN_PLANTID                     TYPE        ZBOT_PLANTID
* | [<-()] RN_ANZAHL                      TYPE        I
* +--------------------------------------------------------------------------------------</SIGNATURE>
  METHOD anz_plant.
    LOOP AT mt_o_plant ASSIGNING FIELD-SYMBOL(<fs_plant>).
      IF <fs_plant>->get_data( )-plantid = in_plantid.
        ADD 1 TO rn_anzahl.
      ENDIF.

    ENDLOOP.
  ENDMETHOD.
```

```
* <SIGNATURE>---------------------------------------------------------------------------------------+
* | Instance Public Method ZBOT_POOL->AUSGABE                                                       |
* +---------------------------------------------------------------------------------------------------+
* +---------------------------------------------------------------------------------------</SIGNATURE>
  METHOD ausgabe.
    DATA: lt_id TYPE TABLE OF zbot_plantid.
    WRITE:/1 'Planzen-Id',
           15 'Planzenname', 40 'Pflanzenfamilie', 60 'Gefundene Anzahl'.
    ULINE.
    LOOP AT mt_o_plant ASSIGNING FIELD-SYMBOL(<fs_plant>).
      IF line_exists( lt_id[ table_line = <fs_plant>->get_data( )-plantid ] ).
      ELSE.
        WRITE:/1 <fs_plant>->get_data( )-plantid,
                15 <fs_plant>->get_data( )-name,
                40 <fs_plant>->get_data( )-famid,
                60 anz_plant( in_plantid = <fs_plant>->get_data( )-plantid ).
        APPEND <fs_plant>->get_data( )-plantid TO lt_id.
      ENDIF.
    ENDLOOP.
  ENDMETHOD.


* <SIGNATURE>---------------------------------------------------------------------------------------+
* | Instance Public Method ZBOT_POOL->ON_PLANT_CREATED                                              |
* +---------------------------------------------------------------------------------------------------+
* | [--->] EO_PLANT                       LIKE                                                      |
* +---------------------------------------------------------------------------------------</SIGNATURE>
  method ON_PLANT_CREATED.
    add_plant( io_plant = eo_plant ).
  endmethod.
ENDCLASS.
```

5. Hauptprogramm

```
*&---------------------------------------------------------------------*
*& Report ZBOT_POOLLISTE
*&---------------------------------------------------------------------*
*&
*&---------------------------------------------------------------------*
REPORT zbot_poolliste.

DATA: go_plant     TYPE REF TO zbot_obj,
      go_plantlist TYPE REF TO zbot_pool,
      gt_fehler    TYPE TABLE OF REF TO zcx_001_bot.

START-OF-SELECTION.
  go_plantlist = NEW #( ).
  SET HANDLER go_plantlist->on_plant_created FOR ALL INSTANCES.
  SELECT * FROM zbot_testdaten INTO TABLE @DATA(gt_testdaten).
  LOOP AT gt_testdaten INTO DATA(gs_testdaten).
    TRY.
        go_plant = NEW zbot_obj( is_plant = gs_testdaten ).
      CATCH zcx_001_bot INTO DATA(go_exc).
        APPEND go_exc TO gt_fehler.
    ENDTRY.
  ENDLOOP.

  WRITE:/ 'Anzahl Pflanzen:', go_plantlist->anzahl( ).
  SKIP 1.
  WRITE:/'Gefundene Pflanzen:'.
  ULINE.
  go_plantlist->ausgabe( ).

  SKIP 1.
  WRITE:/ 'Fehlerliste'.
  ULINE.
  LOOP AT gt_fehler INTO DATA(go_fehler).
    WRITE:/ go_fehler->get_text( ).
  ENDLOOP.
```