\*&---------------------------------------------------------------------\*  
\*& Include          ZFIR\_CASH\_JOURNAL\_SIMPLE\_C01  
  
\*---------------------------------------------------------------------\*  
CLASS lcl\_application\_main IMPLEMENTATION.  
  METHOD show\_alv.  
    init\_alv( CHANGING pt\_fieldcat = gt\_fieldcat  
                       pt\_outtab   = gt\_outtab[] ).  
  ENDMETHOD.                    "show\_alv  
  
  METHOD register\_events.  
    DATA: lt\_f4 TYPE lvc\_t\_f4,  
          ls\_f4 LIKE LINE OF lt\_f4.  
  
\* Register F4 Event.  
    CLEAR lt\_f4.  
  
    ls\_f4-fieldname = 'RECEIPT\_KEY'.  
    ls\_f4-register = abap\_true.  
    ls\_f4-getbefore = abap\_true.  
    ls\_f4-chngeafter = abap\_true.  
    APPEND ls\_f4 TO lt\_f4.  
  
    go\_grid->register\_f4\_for\_fields( EXPORTING it\_f4 = lt\_f4 ).  
  
\* register events for abap objects (backend)  
    CREATE OBJECT go\_onf4.  
    SET HANDLER go\_onf4->on\_f4 FOR go\_grid. " ACTIVATION abap\_true.  
  
\* Register On Enter Event.  
    go\_grid->register\_edit\_event( EXPORTING i\_event\_id = cl\_gui\_alv\_grid=>mc\_evt\_enter ).  
  
  ENDMETHOD.                    "register\_events  
  
  METHOD exclude\_tb\_functions.  
    DATA ls\_exclude TYPE ui\_func.  
  
    REFRESH pt\_exclude.  
  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_loc\_insert\_row.  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_loc\_undo.  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_loc\_copy\_row.  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_sort\_asc.  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_sort\_dsc.  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_loc\_paste\_new\_row.  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_subtot.  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_sum.  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_maximum.  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_minimum.  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_filter .  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_average .  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_print .  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_loc\_cut .  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_loc\_copy .  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_loc\_paste .  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_pc\_file .  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_find .  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_find\_more .  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_detail  .  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_call\_xxl  .  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_check .  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_html.  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_to\_office.  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_view\_excel  .  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_view\_grid  .  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_word\_processor  .  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_send  .  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_call\_abc  .  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_graph  .  
    APPEND ls\_exclude TO pt\_exclude.  
    ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_info  .  
    APPEND ls\_exclude TO pt\_exclude.  
  
    IF gv\_display = abap\_true.  
      ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_loc\_append\_row  .  
      APPEND ls\_exclude TO pt\_exclude.  
      ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_loc\_delete\_row  .  
      APPEND ls\_exclude TO pt\_exclude.  
    ENDIF.  
  ENDMETHOD.                    "exclude\_tb\_functions  
  METHOD build\_fieldcat.  
  
    DATA lv\_out\_len TYPE dd03p-outputlen VALUE '8'.  
  
    FIELD-SYMBOLS: <fs\_fcat> TYPE lvc\_s\_fcat.  
  
    REFRESH: pt\_fieldcat.  
  
    CALL FUNCTION 'LVC\_FIELDCATALOG\_MERGE'  
      EXPORTING  
        i\_structure\_name   = 'ZFI\_CR\_ITEM'  
        i\_bypassing\_buffer = 'X'  
      CHANGING  
        ct\_fieldcat        = pt\_fieldcat  
      EXCEPTIONS  
        OTHERS             = 1.  
  
    CHECK sy-subrc = 0.  
  
    IF gv\_edit = abap\_true OR gv\_create = abap\_true.  
      LOOP AT pt\_fieldcat ASSIGNING <fs\_fcat>.  
        IF <fs\_fcat>-fieldname = 'RECEIPT\_KEY'.  
          <fs\_fcat>-f4availabl = abap\_true.  
          <fs\_fcat>-edit = abap\_true.  
          <fs\_fcat>-outputlen = lv\_out\_len.  
        ENDIF.  
  
        IF <fs\_fcat>-fieldname = 'AMOUNT'.  
          <fs\_fcat>-edit = abap\_true.  
        ENDIF.  
      ENDLOOP.  
    ENDIF.  
  
  ENDMETHOD.                    "build\_fieldcat  
  
  METHOD init\_alv.  
    DATA: lt\_exclude TYPE ui\_functions,  
          ls\_variant TYPE disvariant.  
  
    CREATE OBJECT go\_custom\_container  
      EXPORTING  
        container\_name = g\_container.  
  
    CREATE OBJECT go\_grid  
      EXPORTING  
        i\_parent = go\_custom\_container.  
  
\* Create Objekt to verify input values.  
\* (This object is already needed in form SELECT\_DATA).  
    CREATE OBJECT go\_verifier.  
    SET HANDLER go\_verifier->handle\_data\_changed FOR go\_grid.  
  
\* Build fieldcat and set columns PLANETYPE and SEATSOCC  
\* edit enabled.  
    me->build\_fieldcat( CHANGING pt\_fieldcat = pt\_fieldcat ).  
  
    me->exclude\_tb\_functions( CHANGING pt\_exclude = lt\_exclude ).  
  
    gs\_layout-stylefname = 'CELLTAB'.  
\*    gs\_layout-cwidth\_opt = abap\_true.  
    gs\_layout-col\_opt = abap\_true.  
  
    ls\_variant-report = sy-repid.  
    ls\_variant-username = sy-uname.  
  
  
    CALL METHOD go\_grid->set\_table\_for\_first\_display  
      EXPORTING  
\*       i\_structure\_name     = 'ZFI\_CR\_ITEM'  
        it\_toolbar\_excluding = lt\_exclude  
        is\_layout            = gs\_layout  
        is\_variant           = ls\_variant  
        i\_save               = 'A'  
      CHANGING  
        it\_fieldcatalog      = pt\_fieldcat  
        it\_outtab            = pt\_outtab[].  
  
\* Hide tool bar  
    IF gv\_edit = abap\_false.  
  
\*      CALL METHOD g\_grid->SET\_TOOLBAR\_VISIBLE  
\*        EXPORTING  
\*          VISIBLE = 0.  
    ENDIF.  
  
\* register f4 for field CLASS  
    register\_events( ).  
  
\* Set editable cells to ready for input initially  
    go\_grid->set\_ready\_for\_input( EXPORTING i\_ready\_for\_input = 1 ).  
  
  ENDMETHOD.                    "init\_alv  
ENDCLASS.                    "lcl\_application\_main IMPLEMENTATION  
  
\*----------------------------------------------------------------------\*  
\*       CLASS lcl\_event\_receiver IMPLEMENTATION  
\*----------------------------------------------------------------------\*  
\*  
\*----------------------------------------------------------------------\*  
CLASS lcl\_event\_receiver IMPLEMENTATION.  
  
  METHOD handle\_data\_changed.  
\* Update editable fields.  
    me->update\_editable\_fields( er\_data\_changed ).  
  
  ENDMETHOD.                    "handle\_data\_changed  
  
  METHOD update\_editable\_fields.  
    DATA: ls\_good             TYPE lvc\_s\_modi,  
          lv\_receipt\_key      TYPE z\_receipt\_key,  
          lv\_posting\_key      TYPE bschl,  
          lv\_customer         TYPE kunnr,  
          lv\_country          TYPE land1,  
          lv\_one\_time\_account TYPE xcpdk,  
          lv\_style            TYPE raw4,  
          lv\_koart            TYPE koart,  
          lv\_vendor           TYPE lfa1-lifnr,  
          lv\_fieldname        TYPE lvc\_fname,  
          lv\_no\_val           TYPE abap\_bool.  
  
    DATA: ls\_changed\_data LIKE LINE OF lt\_rec\_buff,  
          lo\_struct\_desc  TYPE REF TO cl\_abap\_structdescr,  
          ls\_default\_val  LIKE LINE OF gt\_zfi\_cr\_def\_val,  
          ls\_def\_item     TYPE lty\_outtab.  
  
    FIELD-SYMBOLS: <fs\_change\_data> LIKE LINE OF lt\_rec\_buff,  
                   <fs\_del\_row>     TYPE lvc\_s\_moce,  
                   <fs\_component>   TYPE abap\_compdescr,  
                   <fs\_val>         TYPE any,  
                   <fs\_cr\_def\_val>  LIKE LINE OF gt\_zfi\_cr\_def\_val.  
  
    BREAK macleos.  
    LOOP AT pr\_data\_changed->mt\_deleted\_rows ASSIGNING <fs\_del\_row>.  
      DELETE lt\_rec\_buff WHERE row\_id = <fs\_del\_row>-row\_id.  
    ENDLOOP.  
  
    LOOP AT pr\_data\_changed->mt\_good\_cells INTO ls\_good.  
      CLEAR: lv\_no\_val.  
  
      CASE ls\_good-fieldname.  
        WHEN 'POSTING\_KEY'.  
          CALL METHOD pr\_data\_changed->get\_cell\_value  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = ls\_good-fieldname  
            IMPORTING  
              e\_value     = lv\_posting\_key.  
          IF lv\_posting\_key IS NOT INITIAL.  
            SELECT SINGLE koart  
              FROM tbsl  
              INTO lv\_koart  
             WHERE bschl = lv\_posting\_key.  
            IF sy-subrc = 0.  
              IF lv\_koart = 'D'.  
                CALL METHOD pr\_data\_changed->modify\_style  
                  EXPORTING  
                    i\_row\_id    = ls\_good-row\_id  
                    i\_fieldname = 'CUSTOMER'  
                    i\_style     = cl\_gui\_alv\_grid=>mc\_style\_enabled.  
  
                CALL METHOD pr\_data\_changed->modify\_style  
                  EXPORTING  
                    i\_row\_id    = ls\_good-row\_id  
                    i\_fieldname = 'REF\_INVOICE'  
                    i\_style     = cl\_gui\_alv\_grid=>mc\_style\_enabled.  
  
\*                call method pr\_data\_changed->modify\_style  
\*                  exporting  
\*                    i\_row\_id    = ls\_good-row\_id  
\*                    i\_fieldname = 'FUND'  
\*                    i\_style     = cl\_gui\_alv\_grid=>mc\_style\_enabled.  
\*  
\*                call method pr\_data\_changed->modify\_style  
\*                  exporting  
\*                    i\_row\_id    = ls\_good-row\_id  
\*                    i\_fieldname = 'GRANT\_NBR'  
\*                    i\_style     = cl\_gui\_alv\_grid=>mc\_style\_enabled.  
  
\*                call method pr\_data\_changed->modify\_style  
\*                  exporting  
\*                    i\_row\_id    = ls\_good-row\_id  
\*                    i\_fieldname = 'BUSINESS\_AREA'  
\*                    i\_style     = cl\_gui\_alv\_grid=>mc\_style\_enabled.  
  
  
              ELSEIF lv\_koart = 'K'.  
                CALL METHOD pr\_data\_changed->modify\_style  
                  EXPORTING  
                    i\_row\_id    = ls\_good-row\_id  
                    i\_fieldname = 'VENDOR'  
                    i\_style     = cl\_gui\_alv\_grid=>mc\_style\_enabled.  
  
                CALL METHOD pr\_data\_changed->modify\_style  
                  EXPORTING  
                    i\_row\_id    = ls\_good-row\_id  
                    i\_fieldname = 'REF\_INVOICE'  
                    i\_style     = cl\_gui\_alv\_grid=>mc\_style\_enabled.  
              ELSEIF lv\_koart = 'S'.  
                CALL METHOD pr\_data\_changed->modify\_style  
                  EXPORTING  
                    i\_row\_id    = ls\_good-row\_id  
                    i\_fieldname = 'CUSTOMER'  
                    i\_style     = cl\_gui\_alv\_grid=>mc\_style\_disabled.  
  
                CALL METHOD pr\_data\_changed->modify\_style  
                  EXPORTING  
                    i\_row\_id    = ls\_good-row\_id  
                    i\_fieldname = 'VENDOR'  
                    i\_style     = cl\_gui\_alv\_grid=>mc\_style\_disabled.  
  
                CALL METHOD pr\_data\_changed->modify\_style  
                  EXPORTING  
                    i\_row\_id    = ls\_good-row\_id  
                    i\_fieldname = 'REF\_INVOICE'  
                    i\_style     = cl\_gui\_alv\_grid=>mc\_style\_disabled.  
              ENDIF.  
            ELSE.  
  
            ENDIF.  
          ENDIF.  
        WHEN 'RECEIPT\_KEY'.  
  
          CALL METHOD pr\_data\_changed->get\_cell\_value  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = ls\_good-fieldname  
            IMPORTING  
              e\_value     = lv\_receipt\_key.  
  
\*         Check if value for receipt key on this row has changed or not  
          READ TABLE lt\_rec\_buff WITH TABLE KEY row\_id = ls\_good-row\_id  
            ASSIGNING <fs\_change\_data>.  
  
\*         We have not encountered the row, add it to the table  
          IF sy-subrc <> 0.  
            ls\_changed\_data-row\_id = ls\_good-row\_id.  
            ls\_changed\_data-receipt\_key = lv\_receipt\_key.  
            INSERT ls\_changed\_data INTO TABLE lt\_rec\_buff.  
  
\*         We only care if the value has changed or if it is blank  
          ELSE.  
            IF <fs\_change\_data>-receipt\_key NE lv\_receipt\_key.  
              DELETE lt\_rec\_buff WHERE row\_id = ls\_good-row\_id.  
              ls\_changed\_data-row\_id = ls\_good-row\_id.  
              ls\_changed\_data-receipt\_key = lv\_receipt\_key.  
              INSERT ls\_changed\_data INTO TABLE lt\_rec\_buff.  
            ELSEIF <fs\_change\_data>-receipt\_key EQ lv\_receipt\_key.  
              CONTINUE.  
            ENDIF.  
          ENDIF.  
  
\*         Check to see if the receipt key exists  
          READ TABLE gt\_zfi\_cr\_def\_val ASSIGNING <fs\_cr\_def\_val>  
                                       WITH KEY receipt\_key = lv\_receipt\_key.  
  
          IF sy-subrc <> 0.  
            IF lv\_receipt\_key IS NOT INITIAL.  
              MESSAGE i013(zfi\_cj) WITH lv\_receipt\_key.  
            ENDIF.  
  
            lv\_style = cl\_gui\_alv\_grid=>mc\_style\_disabled.  
            lv\_no\_val = abap\_true.  
          ELSE.  
            IF <fs\_cr\_def\_val>-editable = abap\_true.  
              lv\_style = cl\_gui\_alv\_grid=>mc\_style\_enabled.  
            ELSE.  
              lv\_style = cl\_gui\_alv\_grid=>mc\_style\_disabled.  
            ENDIF.  
          ENDIF.  
  
\*         When the key is changed we need to initialize the fields within the row  
  
          lo\_struct\_desc ?= cl\_abap\_typedescr=>describe\_by\_name( 'ZFIT\_CR\_DEF\_VAL' ).  
  
          LOOP AT lo\_struct\_desc->components ASSIGNING <fs\_component>.  
            CASE <fs\_component>-name.  
              WHEN 'MANDT' OR 'EDITABLE' OR 'RECEIPT\_KEY'.  
                CONTINUE.  
              WHEN OTHERS.  
                lv\_fieldname = <fs\_component>-name.  
  
                IF lv\_no\_val = abap\_true.  
                  ASSIGN COMPONENT <fs\_component>-name OF STRUCTURE ls\_default\_val TO <fs\_val>.  
                  CHECK sy-subrc = 0.  
                ELSE.  
                  ASSIGN COMPONENT <fs\_component>-name OF STRUCTURE <fs\_cr\_def\_val> TO <fs\_val>.  
                  CHECK sy-subrc = 0.  
                ENDIF.  
  
                CALL METHOD pr\_data\_changed->modify\_cell  
                  EXPORTING  
                    i\_row\_id    = ls\_good-row\_id  
\*                   i\_tabix     =  
                    i\_fieldname = lv\_fieldname  
                    i\_value     = <fs\_val>.  
            ENDCASE.  
          ENDLOOP.  
  
  
\*         When the key is cleared, clear all the values in the cells and initialize them  
          IF lv\_receipt\_key IS INITIAL.  
            lo\_struct\_desc ?= cl\_abap\_typedescr=>describe\_by\_name( 'ZFI\_CR\_ITEM' ).  
  
            LOOP AT lo\_struct\_desc->components ASSIGNING <fs\_component>.  
              IF <fs\_component>-name EQ 'RECEIPT\_KEY' OR <fs\_component>-name EQ 'AMOUNT'.  
                CONTINUE.  
              ENDIF.  
  
              lv\_fieldname = <fs\_component>-name.  
  
              CALL METHOD pr\_data\_changed->modify\_style  
                EXPORTING  
                  i\_row\_id    = ls\_good-row\_id  
                  i\_fieldname = lv\_fieldname  
                  i\_style     = cl\_gui\_alv\_grid=>mc\_style\_disabled.  
  
              ASSIGN COMPONENT <fs\_component>-name OF STRUCTURE ls\_def\_item TO <fs\_val>.  
              CHECK sy-subrc = 0.  
  
              CALL METHOD pr\_data\_changed->modify\_cell  
                EXPORTING  
                  i\_row\_id    = ls\_good-row\_id  
\*                 i\_tabix     =  
                  i\_fieldname = lv\_fieldname  
                  i\_value     = <fs\_val>.  
            ENDLOOP.  
  
            CONTINUE.  
          ENDIF.  
  
\*         BOC TEMP-AVADHUT  
          CALL METHOD pr\_data\_changed->modify\_cell  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
\*             i\_tabix     =  
              i\_fieldname = 'BUDGET\_PD'  
              i\_value     = ''.  
\*         EOC TEMP-AVADHUT  
  
          "Clear on recp key  
          CALL METHOD pr\_data\_changed->modify\_cell  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
\*             i\_tabix     =  
              i\_fieldname = 'AMOUNT'  
              i\_value     = ''.  
  
          CALL METHOD pr\_data\_changed->modify\_cell  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
\*             i\_tabix     =  
              i\_fieldname = 'FKBER'  
              i\_value     = ''.  
  
  
          CALL METHOD pr\_data\_changed->modify\_cell  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
\*             i\_tabix     =  
              i\_fieldname = 'WBS'  
              i\_value     = ''.  
  
          CALL METHOD pr\_data\_changed->modify\_cell  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
\*             i\_tabix     =  
              i\_fieldname = 'CUSTOMER'  
              i\_value     = ''.  
  
          CALL METHOD pr\_data\_changed->modify\_cell  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
\*             i\_tabix     =  
              i\_fieldname = 'REF\_INVOICE'  
              i\_value     = ''.  
  
  
          CALL METHOD pr\_data\_changed->modify\_cell  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
\*             i\_tabix     =  
              i\_fieldname = 'INTERNAL\_ORDER'  
              i\_value     = ''.  
  
          CALL METHOD pr\_data\_changed->modify\_cell  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
\*             i\_tabix     =  
              i\_fieldname = 'FISTL'  
              i\_value     = ''.  
  
          CALL METHOD pr\_data\_changed->modify\_cell  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
\*             i\_tabix     =  
              i\_fieldname = 'SGTXT'  
              i\_value     = ''.  
  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'SPECIAL\_GL\_IND'  
              i\_style     = lv\_style.  
  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'OTA\_DATA'  
              i\_style     = lv\_style.  
  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'COST\_CENTER'  
              i\_style     = lv\_style.  
  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'INTERNAL\_ORDER'  
              i\_style     = lv\_style.  
  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'WBS'  
              i\_style     = lv\_style.  
  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'GL\_ACCOUNT'  
              i\_style     = lv\_style.  
  
\*          if lv\_receipt\_key = '0005'.  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'FUND'  
              i\_style     = lv\_style.  "cl\_gui\_alv\_grid=>mc\_style\_enabled.  
  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'GRANT\_NBR'  
              i\_style     = lv\_style.   "cl\_gui\_alv\_grid=>mc\_style\_enabled.  
  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'BUSINESS\_AREA'  
              i\_style     = lv\_style.   "cl\_gui\_alv\_grid=>mc\_style\_enabled.  
  
          "For New Additional Fields.  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'FKBER'  
              i\_style     = lv\_style.   "cl\_gui\_alv\_grid=>mc\_style\_enabled.  
  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'FISTL'  
              i\_style     = lv\_style.   "cl\_gui\_alv\_grid=>mc\_style\_enabled.  
  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'BUDGET\_PD'  
              i\_style     = lv\_style.   "cl\_gui\_alv\_grid=>mc\_style\_enabled.  
  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'SGTXT'  
              i\_style     = lv\_style.   "cl\_gui\_alv\_grid=>mc\_style\_enabled.  
  
  
\*          endif.  
  
        WHEN 'CUSTOMER'.  
          CALL METHOD pr\_data\_changed->get\_cell\_value  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = ls\_good-fieldname  
            IMPORTING  
              e\_value     = lv\_customer.  
          IF lv\_customer IS NOT INITIAL.  
            SELECT SINGLE xcpdk  
              FROM kna1  
              INTO lv\_one\_time\_account  
             WHERE kunnr = lv\_customer.  
            IF sy-subrc = 0.  
              IF lv\_one\_time\_account = abap\_true.  
                lv\_style = cl\_gui\_alv\_grid=>mc\_style\_enabled.  
              ELSE.  
                lv\_style = cl\_gui\_alv\_grid=>mc\_style\_disabled.  
              ENDIF.  
            ELSE.  
              lv\_style = cl\_gui\_alv\_grid=>mc\_style\_disabled.  
            ENDIF.  
          ELSE.  
            lv\_style = cl\_gui\_alv\_grid=>mc\_style\_disabled.  
          ENDIF.  
  
  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'NAME1'  
              i\_style     = lv\_style.  
  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'NAME2'  
              i\_style     = lv\_style.  
  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'STREET'  
              i\_style     = lv\_style.  
  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'CITY'  
              i\_style     = lv\_style.  
  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'POSTAL\_CODE'  
              i\_style     = lv\_style.  
  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'REGION'  
              i\_style     = lv\_style.  
  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'COUNTRY'  
              i\_style     = lv\_style.  
  
\* Default to country to US if there is no value.  
          CALL METHOD pr\_data\_changed->get\_cell\_value  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'COUNTRY'  
            IMPORTING  
              e\_value     = lv\_country.  
          IF lv\_country IS INITIAL.  
            CALL METHOD pr\_data\_changed->modify\_cell  
              EXPORTING  
                i\_row\_id    = ls\_good-row\_id  
\*               i\_tabix     =  
                i\_fieldname = 'COUNTRY'  
                i\_value     = 'US'.  
  
          ENDIF.  
  
        WHEN 'VENDOR'.  
          CLEAR: lv\_vendor, lv\_one\_time\_account.  
  
          CALL METHOD pr\_data\_changed->get\_cell\_value  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = ls\_good-fieldname  
            IMPORTING  
              e\_value     = lv\_vendor.  
  
          IF lv\_vendor IS NOT INITIAL.  
            SELECT SINGLE xcpdk  
              FROM lfa1  
              INTO lv\_one\_time\_account  
             WHERE lifnr = lv\_vendor.  
  
            IF sy-subrc = 0.  
              IF lv\_one\_time\_account = abap\_true.  
                lv\_style = cl\_gui\_alv\_grid=>mc\_style\_enabled.  
              ELSE.  
                lv\_style = cl\_gui\_alv\_grid=>mc\_style\_disabled.  
              ENDIF.  
            ELSE.  
              lv\_style = cl\_gui\_alv\_grid=>mc\_style\_disabled.  
            ENDIF.  
          ELSE.  
            lv\_style = cl\_gui\_alv\_grid=>mc\_style\_disabled.  
          ENDIF.  
  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'NAME1'  
              i\_style     = lv\_style.  
  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'NAME2'  
              i\_style     = lv\_style.  
  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'STREET'  
              i\_style     = lv\_style.  
  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'CITY'  
              i\_style     = lv\_style.  
  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'POSTAL\_CODE'  
              i\_style     = lv\_style.  
  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'REGION'  
              i\_style     = lv\_style.  
  
          CALL METHOD pr\_data\_changed->modify\_style  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'COUNTRY'  
              i\_style     = lv\_style.  
  
\* Default to country to US if there is no value.  
          CALL METHOD pr\_data\_changed->get\_cell\_value  
            EXPORTING  
              i\_row\_id    = ls\_good-row\_id  
              i\_fieldname = 'COUNTRY'  
            IMPORTING  
              e\_value     = lv\_country.  
  
          IF lv\_country IS INITIAL.  
            CALL METHOD pr\_data\_changed->modify\_cell  
              EXPORTING  
                i\_row\_id    = ls\_good-row\_id  
\*               i\_tabix     =  
                i\_fieldname = 'COUNTRY'  
                i\_value     = 'US'.  
          ENDIF.  
  
      ENDCASE.  
    ENDLOOP.  
  ENDMETHOD.                    "update\_editable\_fields  
  
  METHOD clear\_rec\_buffer.  
    CLEAR lt\_rec\_buff.  
  ENDMETHOD.                    "clear\_rec\_buffer  
\*------------------------------------------------------  
ENDCLASS.                    "lcl\_event\_receiver IMPLEMENTATION  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
\* LOCAL CLASSES: Implementation  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
\*===============================================================  
\* class lcl\_application\_f4 (Implementation)  
\*  
CLASS lcl\_application\_f4 IMPLEMENTATION.  
  
\*§2. Implement an event handler method for event ONF4.  
  
  METHOD on\_f4.  
  
    DATA: lt\_return     TYPE tfw\_ddshretval\_tab,  
          lv\_rec\_key    TYPE zfit\_cr\_def\_val-receipt\_key,  
          lv\_fieldvalue TYPE help\_info-fldvalue.  
  
    FIELD-SYMBOLS: <fs\_return> LIKE LINE OF lt\_return.  
  
\* Save event parameter as global attributes of this class  
\* (maybe solved differently if you use a function module!)  
    f4\_params-c\_fieldname = e\_fieldname.  
    f4\_params-cs\_row\_no = es\_row\_no.  
    f4\_params-cr\_event\_data = er\_event\_data.  
    f4\_params-ct\_bad\_cells = et\_bad\_cells.  
    f4\_params-c\_display = e\_display.  
  
    IF gv\_display = abap\_true.  
      er\_event\_data->m\_event\_handled = abap\_true.  
      RETURN.  
    ENDIF.  
  
    lv\_fieldvalue = e\_fieldvalue.  
  
    CALL FUNCTION 'F4IF\_FIELD\_VALUE\_REQUEST'  
      EXPORTING  
        tabname           = space  
        fieldname         = space  
        searchhelp        = 'ZFI\_RECEIPT\_KEY'  
        value             = lv\_fieldvalue  
      TABLES  
        return\_tab        = lt\_return  
      EXCEPTIONS  
        field\_not\_found   = 1  
        no\_help\_for\_field = 2  
        inconsistent\_help = 3  
        no\_values\_found   = 4  
        OTHERS            = 5.  
  
    IF sy-subrc <> 0.  
      RETURN.  
    ENDIF.  
  
    IF lines( lt\_return ) > 0.  
      READ TABLE lt\_return ASSIGNING <fs\_return> INDEX 1.  
      lv\_rec\_key = <fs\_return>-fieldval.  
  
      me->populate\_alv( lv\_rec\_key ).  
    ENDIF.  
  
\*§7. Inform the ALV Grid Control that an own f4 help has been processed  
\*    to suppress the standard f4 help.  
    er\_event\_data->m\_event\_handled = abap\_true.  
  
  ENDMETHOD.                                                "on\_f4  
  
  METHOD populate\_alv.  
  
    DATA: ls\_modi          TYPE lvc\_s\_modi,  
          lv\_baseline\_date TYPE char10.  
  
    FIELD-SYMBOLS: <fs\_itab>    TYPE lvc\_t\_modi,  
                   <fs\_def\_val> TYPE zfit\_cr\_def\_val.  
  
\*\*§  6b. Dereference attribute M\_DATA into your field symbol and add  
\*\*       the selected value to the table to which this symbol points to.  
    ASSIGN f4\_params-cr\_event\_data->m\_data->\* TO <fs\_itab>.  
    CHECK sy-subrc = 0.  
  
    ls\_modi-row\_id = f4\_params-cs\_row\_no-row\_id.  
  
    READ TABLE gt\_zfi\_cr\_def\_val ASSIGNING <fs\_def\_val> WITH KEY receipt\_key = iv\_rec\_key.  
    CHECK sy-subrc = 0.  
  
    ls\_modi-fieldname = 'RECEIPT\_KEY'.  
    ls\_modi-value = <fs\_def\_val>-receipt\_key.  
    APPEND ls\_modi TO <fs\_itab>.  
  
    ls\_modi-fieldname = 'RECEIPT\_KEY\_DESC'.  
    ls\_modi-value = <fs\_def\_val>-receipt\_key\_desc.  
    APPEND ls\_modi TO <fs\_itab>.  
  
    ls\_modi-fieldname = 'POSTING\_KEY'.  
    ls\_modi-value = <fs\_def\_val>-posting\_key.  
    APPEND ls\_modi TO <fs\_itab>.  
  
    ls\_modi-fieldname = 'GL\_ACCOUNT'.  
    ls\_modi-value = <fs\_def\_val>-gl\_account.  
    APPEND ls\_modi TO <fs\_itab>.  
  
    ls\_modi-fieldname = 'SPECIAL\_GL\_IND'.  
    ls\_modi-value = <fs\_def\_val>-special\_gl\_ind.  
    APPEND ls\_modi TO <fs\_itab>.  
  
    ls\_modi-fieldname = 'COST\_CENTER'.  
    ls\_modi-value = <fs\_def\_val>-cost\_center.  
    APPEND ls\_modi TO <fs\_itab>.  
  
    ls\_modi-fieldname = 'FUND'.  
    ls\_modi-value = <fs\_def\_val>-fund.  
    APPEND ls\_modi TO <fs\_itab>.  
  
    ls\_modi-fieldname = 'GRANT\_NBR'.  
    ls\_modi-value = <fs\_def\_val>-grant\_nbr.  
    APPEND ls\_modi TO <fs\_itab>.  
  
    ls\_modi-fieldname = 'BUSINESS\_AREA'.  
    ls\_modi-value = <fs\_def\_val>-business\_area.  
    APPEND ls\_modi TO <fs\_itab>.  
  
\* Convert Dates to proper format  
    CALL FUNCTION 'CONVERSION\_EXIT\_PDATE\_OUTPUT'  
      EXPORTING  
        input  = <fs\_def\_val>-baseline\_date  
      IMPORTING  
        output = lv\_baseline\_date.  
  
    ls\_modi-fieldname = 'BASELINE\_DATE'.  
    ls\_modi-value = lv\_baseline\_date.  
    APPEND ls\_modi TO <fs\_itab>.  
    UNASSIGN <fs\_itab>.  
  
  ENDMETHOD.                    "populate\_alv  
  
ENDCLASS.                    "lcl\_application\_f4 IMPLEMENTATION  
\*&---------------------------------------------------------------------\*  
\*&      Module  F4\_DOC\_TYPE  INPUT  
\*&---------------------------------------------------------------------\*  
\*       text  
\*----------------------------------------------------------------------\*  
MODULE f4\_doc\_type INPUT.  
  TYPES: BEGIN OF ty\_t003t,  
           blart TYPE blart,  
           ltext TYPE ltext\_003t,  
         END OF ty\_t003t.  
  
  DATA: ls\_dynpread TYPE dynpread,  
        lt\_dynpread TYPE dynpread\_t,  
        lt\_return   TYPE tinv\_ddshretval,  
        ls\_t003     TYPE ty\_t003t,  
        lt\_t003     TYPE STANDARD TABLE OF ty\_t003t.  
  
  CLEAR: lt\_dynpread.  
  
  SELECT blart, ltext  
    FROM t003t INTO TABLE @lt\_t003  
    WHERE spras = @sy-langu.  
  
  
  CALL FUNCTION 'F4IF\_INT\_TABLE\_VALUE\_REQUEST'  
    EXPORTING  
\*     DDIC\_STRUCTURE  = ' '  
      retfield        = 'BLART'  
\*     PVALKEY         = ' '  
      dynpprog        = sy-repid  
      dynpnr          = sy-dynnr  
      dynprofield     = '0100\_I\_DOCUMENT\_TYPE'  
\*     stepl           = lv\_dynindex  
\*     WINDOW\_TITLE    =  
\*     VALUE           = ' '  
      value\_org       = 'S'  
\*     MULTIPLE\_CHOICE = ' '  
\*     DISPLAY         = ' '  
\*     CALLBACK\_PROGRAM       = ' '  
\*     CALLBACK\_FORM   = ' '  
\*     CALLBACK\_METHOD =  
\*     MARK\_TAB        =  
\*   IMPORTING  
\*     USER\_RESET      =  
    TABLES  
      value\_tab       = lt\_t003  
\*     FIELD\_TAB       =  
      return\_tab      = lt\_return  
\*     DYNPFLD\_MAPPING =  
    EXCEPTIONS  
      parameter\_error = 1  
      no\_values\_found = 2  
      OTHERS          = 3.  
  IF sy-subrc <> 0.  
\* Implement suitable error handling here  
  ENDIF.  
  IF NOT lt\_return IS INITIAL.  
    TRY.  
        DATA(lv\_return) = lt\_return[ 1 ]-fieldval.  
      CATCH cx\_sy\_itab\_line\_not\_found.  
        RETURN.  
    ENDTRY.  
  ENDIF.  
  
  ls\_t003-blart = lv\_return.  
  
  ls\_dynpread-fieldname = '0100\_I\_DOCUMENT\_TYPE'.  
  ls\_dynpread-fieldvalue = ls\_t003-blart.  
  APPEND ls\_dynpread TO lt\_dynpread.  
  
  
  CALL FUNCTION 'DYNP\_VALUES\_UPDATE'  
    EXPORTING  
      dyname               = sy-repid  
      dynumb               = sy-dynnr  
    TABLES  
      dynpfields           = lt\_dynpread  
    EXCEPTIONS  
      invalid\_abapworkarea = 1  
      invalid\_dynprofield  = 2  
      invalid\_dynproname   = 3  
      invalid\_dynpronummer = 4  
      invalid\_request      = 5  
      no\_fielddescription  = 6  
      undefind\_error       = 7  
      OTHERS               = 8.  
  
  IF sy-subrc <> 0.  
\* Implement suitable error handling here  
  ENDIF.  
ENDMODULE.