

Chapter 34: Branching to List Processing

Overview

Often you want to produce a list from within your transaction. There are two ways you can do this.

- Submit a separate report
Use the SUBMIT statement to start a separate report directly from the transaction. Using the SUBMIT statement is described in *Submitting Reports*.
- Produce the list from your module pool using LEAVE TO LIST-PROCESSING

The LEAVE TO LIST-PROCESSING statement is the statement you use to produce a list from a module pool. This statement lets you switch from dialog-mode into list-mode within a dialog program. You can then code the list-processing logic you need right in the module pool.

At runtime, when a LEAVE TO LIST-PROCESSING statement is executed, the module pool still retains control of execution. The transaction's data area is completely available to the report-processing code, so no parameters must be passed back and forth.

When you branch into list-mode, you can produce lists using all the ABAP/4 features available for interactive reporting. This is particularly useful during PROCESS ON VALUE-REQUEST or PROCESS ON HELP-REQUEST processing, when you want to customize the display of field help or possible values.

See the following for information:

Using LEAVE TO LIST-PROCESSING

Using GUI Statuses in List-Mode

Returning to Dialog-Mode

For a concrete example of how to branch into list-mode, see transaction TZ70 (in development class SDWA, which is delivered with your system) and the discussion of it in this chapter:

Example Transaction: Branching to List-Mode

Contents

Using LEAVE TO LIST-PROCESSING	34-2
Using GUI Statuses in List-Mode.....	34-3
Returning to Dialog-Mode.....	34-4

Error! No text of specified style in document.

Using LEAVE TO LIST-PROCESSING

To use LEAVE TO LIST-PROCESSING, place the statement in the code where you want list-mode processing to begin. This statement does two things:

- Switches to list-mode processing
From this point on, you can code the reporting statements needed to issue and control the report. All the standard report-related events and features are available: AT LINE-SELECTION, function keys, basic and detail list levels, windows and so on.
- Sets the standard list output to be the “following screen” for the current screen
The system notes that the standard list-output should follow the display of the current screen. Depending on your application, you may want both displays, or you can inhibit the current screen and replace it with the list output.

Very often, it is cleaner to enclose all reporting code in a single subroutine. Example transaction TZ70 does this:

**** ABAP/4 module and form: ****

```
MODULE PREPARE_LIST OUTPUT.
  LEAVE TO LIST-PROCESSING AND RETURN TO SCREEN 0.
  PERFORM EDIT_LIST.
  LEAVE SCREEN.
ENDMODULE.

FORM EDIT_LIST.
  SET PF-STATUS 'LIST'.
  SET TITLEBAR 'LST' WITH SFLIGHT-CONNID SFLIGHT-CARRID.

  NEW-PAGE LINE-SIZE 72.
  SELECT * FROM SFLIGHT WHERE      CARRID = SFLIGHT-CARRID
                                AND CONNID = SFLIGHT-CONNID.
    WRITE: / SY-VLINE NO-GAP,
            SFLIGHT-FLDATE   COLOR 4 INTENSIFIED OFF NO-GAP,
            SY-VLINE NO-GAP,
            SFLIGHT-PRICE    COLOR 2 INTENSIFIED OFF NO-GAP,
            .....
ENDFORM.
```

How List-Mode in Dialog-Mode Works

At runtime, the module pool retains control of execution. You can code list-mode logic in PBO or PAI for the current screen. The choice depends on whether you want the list output to follow the current screen, or to replace it. In either case, the list appears when processing for the current screen terminates. (Screen processing terminates when control reaches either a LEAVE SCREEN statement or the end of PAI.)

- To display the list output *in addition to* the current screen:
Place the LEAVE TO LIST-PROCESSING logic at the end of PAI. Coded this way, the program can respond to requests for list output within the current PAI processing. On return from the list display, the system repeats processing for the current screen, starting with the beginning of PBO.
- To display the list output *instead of* the current screen:

Code the LEAVE TO LIST-PROCESSING logic in the PBO, and follow it with LEAVE SCREEN. This tells the system to display the list without displaying the current screen. PAI processing for the current screen is not executed.

For more information, see:

Processing Screens in Background

Example Transaction: Branching to List-Mode

Using GUI Statuses in List-Mode

You can use the standard GUI status for lists, or you can define your own interface. If you use the standard list status, the system automatically implements many GUI functions for you. For example:

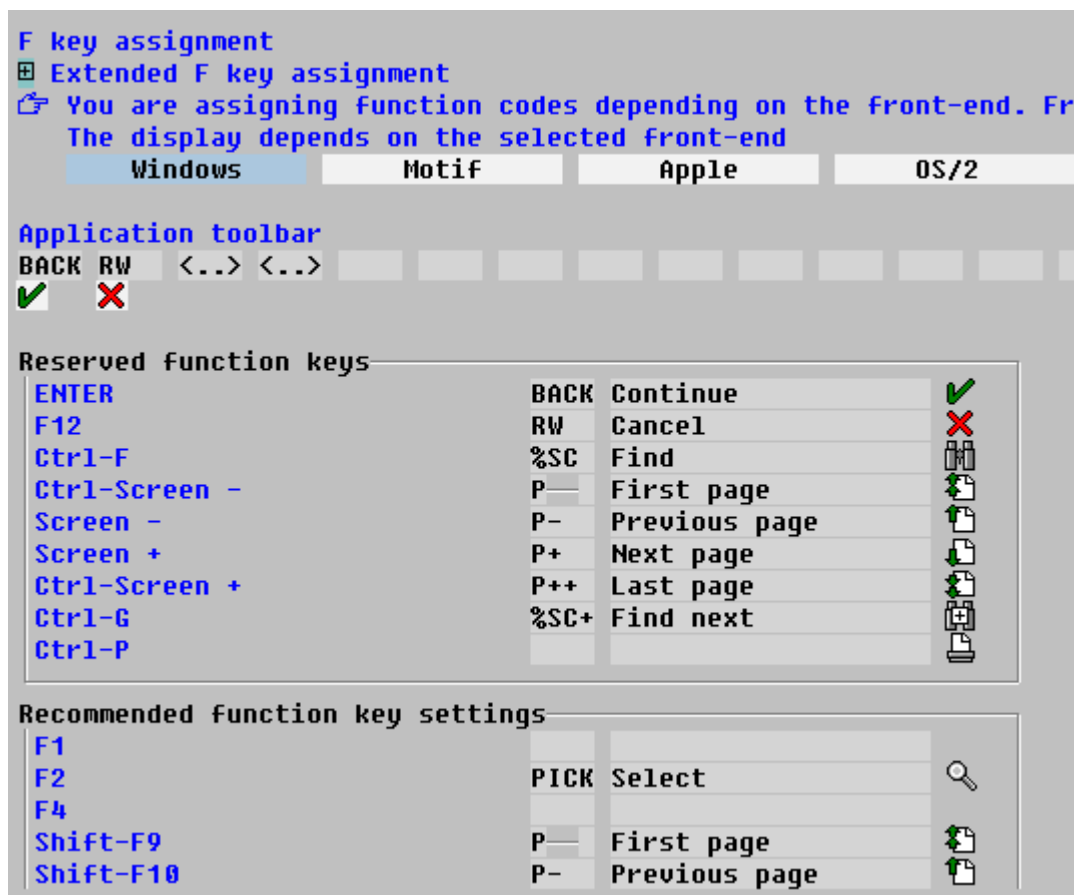
- two return functions (BACK, CANCEL)
- the *Print* and *Search* functions
- the standard scrolling keys (P+, P++, P-, P--)

You do not need to program these functions in your own code.

If you define your own status, you must do two things. First, activate the desired functions explicitly in the GUI status (by entering their function codes in the relevant pushbuttons and function codes). Second, in your ABAP/4 module, code the logic for those functions not handled automatically by the system. This includes the **F21-F24** keys, although the P-, P-, P+, and P++ functions are automatically defined.

Example transaction TZ70 defines its own status (named LIST). Since this status has type *List in dialog box*, only pushbuttons can be defined. (A menu bar with menus is not needed). The *Back* and *Cancel* exit functions (BACK und RW) appear automatically as pushbuttons, because the system implements these functions itself.

Error! No text of specified style in document.



Note

For a list of all functions in list-mode that are handled automatically by the system, do the following:

1. In the Menu Painter worksheet, select *Utilities* → *Explanations* → *Norms/suggestions*.
2. In the resulting popup, double-click on the *List functions* element of the hierarchy.

For information on interactive reporting techniques, see *Interactive Lists*.

Returning to Dialog-Mode

There are two ways to return from list-mode to dialog-mode. In both cases, the place your program returns to is, by default, the screen that started list processing. (This is the screen containing the LEAVE TO LIST-PROCESSING statement.)

If you want to return to a different screen, see *Returning to a Different Screen*.

Your program continues to run in list-mode until one of the following occurs:

- The system reaches a LEAVE LIST-PROCESSING statement in your code.
The LEAVE LIST-PROCESSING statement returns control to the dialog screen. On return, the system re-starts processing at the beginning of PBO.

- The user requests **BACK** or **CANCEL** from the basic-list level of the report.

If the user exits the list using the **BACK** or **CANCEL** icons, you do not need to program an explicit **LEAVE LIST-PROCESSING**. **BACK** or **CANCEL** are standard return functions that are automatically handled by the system. When the user presses one of these, the system returns to the screen containing the **LEAVE TO LIST-PROCESSING**. Here the system re-starts PBO processing screen.

(Remember that the **EXIT** function, unlike **BACK** or **CANCEL**, is *not* handled by the system. If you want to offer the **EXIT** function, you must implement it yourself. In this case, implement it using a **LEAVE LIST-PROCESSING** statement.)

Example transaction TZ70 returns from list-mode using the second method. For screen 200, where the transaction branches into list mode, no exit functions are coded, since **BACK** and **CANCEL** are offered automatically by the system. For screen 100 however, where there is no list processing, the program must implement its own exit functions (**BACK**, **EXIT**, and **CANCEL**):

```
MODULE EXIT_0100 INPUT.
  CASE OK_CODE.
    WHEN 'CANC'.
      CLEAR OK_CODE.
      SET SCREEN 0. LEAVE SCREEN.
    WHEN 'EXIT'.
      CLEAR OK_CODE.
      SET SCREEN 0. LEAVE SCREEN.
    WHEN 'BACK'.
      CLEAR OK_CODE.
      SET SCREEN 0. LEAVE SCREEN.
  ENDCASE.
ENDMODULE.
```

Returning to a Different Screen

When returning to dialog-mode, your program can also re-route the user to a screen different from the one that started the list. To do this, use the keywords **AND RETURN TO SCREEN** when you first branch to list-mode:

```
LEAVE TO LIST-PROCESSING AND RETURN TO SCREEN 100.
```

With this statement, whenever the program returns to dialog mode (either because the user exits from a list, or because a **LEAVE LIST-PROCESSING** is executed), the system resumes processing at the PBO for the screen requested (here screen 100).

Transaction TZ70 uses **AND RETURN TO SCREEN 0** in the **PREPARE_LIST** module:

```
** PROCESSING FOR SCREEN 200**
MODULE PREPARE_LIST OUTPUT.
  LEAVE TO LIST-PROCESSING AND RETURN TO SCREEN 0.
  PERFORM EDIT_LIST.
  LEAVE SCREEN.
ENDMODULE.
```

Here, returning to screen 0 closes the **CALL** mode started when screen 200 was triggered with **CALL SCREEN**. The system returns control to the last screen in the previous call mode (that is, in the PBO for screen 100).

Error! No text of specified style in document.

Example Transaction: Branching to List-Mode

Transaction TZ70 demonstrates one way to perform list processing inside a transaction.:

1. Screen 100 PAI: CALL SCREEN 200.
2. Screen 200 PBO: leave to list-processing, generate list, and leave screen 200.
 - No display of Screen 200 (inhibited by LEAVE SCREEN statement)
3. Screen 200 PAI: no coding needed (inhibited by LEAVE SCREEN)
 - Display of list output occurs:

Airline carrier

Flight number

Flights for flight number 0017 company AA					
Date	Ticket price	Curr	Jet	Capacity	Occupied
30.01.1995	689,66	USD	747-400	660	10
01.02.1995	689,66	USD	747-400	660	20
01.06.1995	689,66	USD	747-400	660	38
04.06.1995	689,66	USD	747-400	660	38

☐ ☐

- Return to PBO for screen 100 (because user presses either *Cancel* or *End*)
4. Screen 100 PBO: re-execute PBO...

Screen Flow Logic

The relevant parts of the flow logic for the two screens are:

```

*-----*
*   Screen 100: Flow Logic   (PAI only)   *
*&-----*
PROCESS AFTER INPUT.
  MODULE EXIT_0100 AT EXIT-COMMAND.
  MODULE USER_COMMAND_0100.

*-----*
*   Screen 200: Flow Logic   *
*&-----*
PROCESS BEFORE OUTPUT.
  MODULE PREPARE_LIST.
*
PROCESS AFTER INPUT.

```

ABAP/4 Code

The main PAI module for screen 100 calls screen 200, which will appear as a separate dialog box (popup):

```
*&-----*
*&      Module  USER_COMMAND_0100  INPUT
*&-----*
MODULE USER_COMMAND_0100 INPUT.
  CASE OK_CODE.
    WHEN SPACE.
      <...Select flight info for the requested flight..>
      CLEAR OK_CODE.
      CALL SCREEN 200 STARTING AT 10 5 ENDING AT 80 15.
    ENDCASE.
  ENDMODULE.
```

The PBO module for screen 200 sets up the branch to list-mode:

```
*&-----*
*&      Module  PREPARE_LIST  OUTPUT
*&-----*
MODULE PREPARE_LIST OUTPUT.
  LEAVE TO LIST-PROCESSING AND RETURN TO SCREEN 0.
  PERFORM EDIT_LIST.
  LEAVE SCREEN.
ENDMODULE.
```

The subroutine for putting out the list contains regular report processing (shown partially):

```
*&-----*
*&      Form  EDIT_LIST
*&-----*
FORM EDIT_LIST.
  SET PF-STATUS 'LIST'.
  SET TITLEBAR 'LST' WITH SFLIGHT-CONNID SFLIGHT-CARRID.

  NEW-PAGE LINE-SIZE 72.
  SELECT * FROM SFLIGHT WHERE      CARRID = SFLIGHT-CARRID
                        AND CONNID = SFLIGHT-CONNID.
    WRITE: /      SY-VLINE NO-GAP,
            SFLIGHT-FLDATE      COLOR 4 INTENSIFIED OFF NO-GAP,
*            .....
  ENDSELECT.
  IF SY-SUBRC = 0. ULINE. ENDIF.
ENDFORM.
```

The list-mode event TOP_OF_PAGE is coded for the list display:

```
TOP-OF-PAGE.
  ULINE.
  WRITE: /      SY-VLINE NO-GAP,
          .....

```