SELECTION SCREENS AND LIST PROCESSING EVENTS

Getting Data from User at Runtime

PARAMETERS is the most basic type of input acceptance.

Using Parameters generates a default selection screen—screen 1000.

If the type of item prompted is based on a global type this allows input help (F1) and entry selection help (F4) are shown as defined in the ABAP dictionary.

PARAMETERS: carrier TYPE sflight-carrid,
              fltdate TYPE sflight-fltdate.
Event blocks

An ABAP program is a collection of processing blocks.

**Event blocks**—called by runtime environment when related event has been triggered through natural program flow of control or when triggered by a user action.

Our initial programs—one block—so no need to declare event blocks.

Program event blocks:

- **LOAD-OF-PROGRAM** (called at initial program setup)
- **START-OF-SELECTION** (beginning of main processing)

  Example use: PARAMETERS triggered at beginning of main processing. If want to set a default value for field based on processing (calculation, data retrieval), must do this before START-OF-SELECTION.

### Example

```plaintext
PARAMETERS date LIKE sy-datum DEFAULT sy-datum.
WRITE date.

DATA pastdate LIKE sy-datum.
pastdate = pastdate - 7.
PARAMETERS date LIKE sy-datum DEFAULT pastdate.
WRITE date.

LOAD-OF-PROGRAM.
DATA pastdate LIKE sy-datum.
pastdate = sy-datum - 7.
PARAMETERS date LIKE sy-datum DEFAULT pastdate.

START-OF-SELECTION.
WRITE date.
```
Selection Screen Input

Selection screen display and control can incorporate additional elements beyond PARAMETERS to provide additional GUI controls. Covered previously:

Selection Texts (changes prompting associated with PARAMETERS).

More on Parameters--checkboxes

PARAMETERS can be designated AS CHECKBOX.

PARAMETERS: myvar AS CHECKBOX.

Use DEFAULT 'X' to have it toggled on initially.

PARAMETERS: myvar AS CHECKBOX DEFAULT 'X'.

Test if user selected by seeing if variable set to 'X'.

IF myvar = 'X'.
"code here.
ENDIF.
More on Parameters--checkboxes

PARAMETERS: male AS CHECKBOX, female AS CHECKBOX.

IF male = 'X'.
  WRITE / 'You are male!'.
ENDIF.

IF female = 'X'.
  WRITE / 'You are female!'.
ENDIF.

More on Parameters--radio buttons

PARAMETERS can be designated as part of a radiobutton group.

PARAMETERS: choice1 RADIOBUTTON GROUP grpa,
  choice2 RADIOBUTTON GROUP grpa,
  choice3 RADIOBUTTON GROUP grpa.

Maximum group name length--4 characters

Use DEFAULT 'X' to have it toggled on initially.

PARAMETERS: choice1 RADIOBUTTON GROUP grpa DEFAULT 'X'.

Test if user selected by seeing if variable set to 'X' or use a CASE
  statement.

CASE 'X'.
  WHEN choice1.
  WHEN choice2.
...
ENDCASE.
More on Parameters--radio buttons

PARAMETERS: male RADIOBUTTON GROUP gen,
              female RADIOBUTTON GROUP gen.

IF male = 'X'.
   WRITE / 'You are male!'.
ENDIF.
IF female = 'X'.
   WRITE / 'You are female!'.
ENDIF.

PARAMETERS: male RADIOBUTTON GROUP gen,
              female RADIOBUTTON GROUP gen.

CASE 'X'.
   WHEN male.
      WRITE / 'You are male!'.
   WHEN female.
      WRITE / 'You are female!'.
ENDCASE.

PARAMETERS: myvar AS CHECKBOX.
PARAMETERS: choice1 RADIOBUTTON GROUP grpa,
            choice2 RADIOBUTTON GROUP grpa,
            choice3 RADIOBUTTON GROUP grpa.

IF myvar = 'X'.
   WRITE / 'You chose the checkbox'.
ELSE.
   WRITE / 'You did not choose the checkbox'.
ENDIF.

CASE 'X'.
   WHEN choice1.
      WRITE / 'You picked choice1'.
   WHEN choice2.
      WRITE / 'You picked choice2'.
   WHEN choice3.
      WRITE / 'You picked choice3'.
   WHEN OTHERS.
      WRITE / 'You did not pick a choice!'.
ENDCASE.
Selection Screen refinements

To exert further control over selection screen use SELECTION-SCREEN statement.

Can add 1 or more framed blocks with optional title.

```plaintext
SELECTION-SCREEN BEGIN OF BLOCK name WITH FRAME TITLE title.
PARAMETERS: male RADIOBUTTON GROUP gen,
            female RADIOBUTTON GROUP gen.
SELECTION-SCREEN END OF BLOCK name.
CASE 'X'.
  WHEN male.
    WRITE / 'You are male!'.
  WHEN female.
    WRITE / 'You are female!'.
ENDCASE.
LOAD-OF-PROGRAM.
title = 'Choose a gender'.
```

Multiple items on a line

To place multiple items on a single output line, place control definitions within a SELECTION-SCREEN BEGIN OF LINE/ SELECTION-SCREEN END OF LINE block.

```plaintext
SELECTION-SCREEN BEGIN OF BLOCK name WITH FRAME TITLE title.
  SELECTION-SCREEN BEGIN OF LINE.
    PARAMETERS: var1 TYPE i,
                var2 TYPE i,
                var3 TYPE i.
  SELECTION-SCREEN END OF LINE.
SELECTION-SCREEN END OF BLOCK name.
LOAD-OF-PROGRAM.
title = 'Enter choices'.
```
COMMENT can be used to place text on the form.
Must use column and length format specifier.

```
SELECTION-SCREEN BEGIN OF BLOCK name WITH FRAME TITLE title.
  SELECTION-SCREEN BEGIN OF LINE.
    SELECTION-SCREEN COMMENT 1(40) comment.
    PARAMETERS: var1 TYPE i,
                var2 TYPE i,
                var3 TYPE i.
  SELECTION-SCREEN END OF LINE.
SELECTION-SCREEN END OF BLOCK name.

LOAD-OF-PROGRAM.
  title = 'Enter choices'.
  comment = 'Enter your three favorite numbers'.
```

```
SELECTION-SCREEN BEGIN OF BLOCK name WITH FRAME TITLE title.
  SELECTION-SCREEN BEGIN OF LINE.
    SELECTION-SCREEN COMMENT 1(5) comment.
    PARAMETERS: male RADIOBUTTON GROUP gen.
    PARAMETERS: female RADIOBUTTON GROUP gen.
  SELECTION-SCREEN END OF LINE.
SELECTION-SCREEN END OF BLOCK name.

CASE 'X'.
  WHEN male.
    WRITE / 'You are male!'.
  WHEN female.
    WRITE / 'You are female!'.
ENDCASE.

LOAD-OF-PROGRAM.
  title = 'Choose a gender'.
  comment = 'Male'.
  comment2 = 'Female'.
```
Allowing range of values selection

Allows user to select a range of values.

\textbf{SELECT-OPTIONS itabname FOR dataobject.}

User entry stored in internal table \texttt{itabname}. 4 fields:

- \textbf{Sign}—either I or E—indicating whether values specified are included or excluded from selection
- \textbf{Low}—the lower limit or a single value (if only one specified)
- \textbf{High}—the upper limit
- \textbf{Option}—text such as EQ, NE, GE, GT, LE, LT (if only Low specified). [Other options not presented here]

Control typically used with database query operations and WHERE clause particularly adjusted to correspond with this resource.

\begin{verbatim}
SELECT fields FROM table WHERE value IN itabname.
\end{verbatim}

(Used as a field limiter \textbf{if selected}, so if not populated all values returned.)

---

\textbf{DATA str TYPE spfli.}

\textbf{SELECT-OPTIONS userair FOR str-carrid.}

\begin{verbatim}
SELECT carrid connid cityfrom cityto FROM spfli INTO CORRESPONDING FIELDS OF str WHERE carrid IN userair.
\end{verbatim}

\textbf{ENDSELECT.}
SELECTION-SCREEN BEGIN OF BLOCK name WITH FRAME TITLE title.
SELECTION-SCREEN BEGIN OF LINE.
  SELECTION-SCREEN COMMENT 1(40) comment.
  PARAMETERS: var1 TYPE i,
             var2 TYPE i.
SELECTION-SCREEN END OF LINE.

SELECTION-SCREEN BEGIN OF LINE.
SELECTION-SCREEN COMMENT 1(40) comment2.
PARAMETERS var3 TYPE i.
SELECTION-SCREEN END OF LINE.
SELECTION-SCREEN END OF BLOCK name.

LOAD-OF-PROGRAM.
title = 'Enter choices'.
comment = 'Enter your three favorite numbers'.
comment2 = 'Put your most favorite here'.

SELECTION-SCREEN BEGIN OF BLOCK name WITH FRAME TITLE title.
SELECTION-SCREEN BEGIN OF LINE.
  SELECTION-SCREEN COMMENT 1(40) comment.
  PARAMETERS: var1 TYPE i.
SELECTION-SCREEN COMMENT 60(1) comment3.
  PARAMETERS: var2 TYPE i.
SELECTION-SCREEN END OF LINE.

SELECTION-SCREEN BEGIN OF LINE.
SELECTION-SCREEN COMMENT 1(40) comment2.
PARAMETERS var3 TYPE i.
SELECTION-SCREEN END OF LINE.
SELECTION-SCREEN END OF BLOCK name.

LOAD-OF-PROGRAM.
title = 'Enter choices'.
comment = 'Enter your three favorite numbers'.
comment2 = 'Put your most favorite here'.
Data Validity Testing

Validation of screen input can be done on the AT SELECTION-SCREEN event.
Useful if want to require user data entry before main processing.

AT SELECTION-SCREEN ON fieldname.
    Triggered if fieldname was submitted.
    If an error or warning is issued, the field is highlighted.

AT SELECTION-SCREEN ON RADIOBUTTON GROUP grpnam.
    If an error or warning is issued, the group is highlighted.

Data Validity Testing

SELECTION-SCREEN BEGIN OF BLOCK name WITH FRAME TITLE title.
    SELECTION-SCREEN BEGIN OF LINE.
    SELECTION-SCREEN COMMENT 1(40) comment.
    PARAMETERS: var1 TYPE i, var2 TYPE i, var3 TYPE i.
    SELECTION-SCREEN END OF LINE.
SELECTION-SCREEN END OF BLOCK name.

AT SELECTION-SCREEN ON var2.
    IF var2 = 7.
        MESSAGE 'Sevens are bad!' TYPE 'W'.
    ENDIF.
List output events

We now move from input and related events to output and a related event.

**AT LINE-SELECTION EVENT**

When outputting a list, the **AT LINE-SELECTION event** is triggered when a line is double clicked or the user chooses the *choose* icon. SY-LSIND starts out at 0 for the initial list output and is incremented each time the event fires. It is decremented as the user moves BACK.

```plaintext
START-OF-SELECTION.
WRITE: /'hello', 'sy-lsind', sy-lsind.

AT LINE-SELECTION.
WRITE: /'Now at selection level', sy-lsind.
```

When writing in AT LINE-SELECTION, this is called a **detail list** and its output appears as a new list.
Retrieving values from selected line

When list is output, HIDE can be used to designate values that will be passed on (called a data transport) when line is selected. The data object to be hidden may or may not be a part of the output that is displayed.

```abap
TYPES spfli_table_type TYPE STANDARD TABLE OF spfli.
DATA itable_spfli TYPE spfli_table_type.
DATA str LIKE LINE OF itable_spfli.
DATA str2 TYPE sflight.

SELECT carrid connid airpfrom airpto deptime FROM spfli INTO CORRESPONDING FIELDS OF TABLE itable_spfli.

LOOP AT itable_spfli INTO str.
  HIDE: str-carrid, str-connid.
ENDLOOP.

AT LINE-SELECTION.
  IF sy-lsind = 1.
    WRITE: 'Flights for connection', str-carrid, str-connid.
    SELECT fldate seatsmax seatsocc FROM sflight INTO CORRESPONDING FIELDS OF str2 WHERE carrid = str-carrid AND connid = str-connid.
    WRITE: / str2-fldate, str2-seatsmax, str2-seatsocc.
  ENDSELECT.
ENDIF.
```