ABAP Part I

Lesson 10: Reporting Events

Lesson Objectives

After completing this lesson, participants will be able to -

- Identify the different events to trigger the Interactive report.
- Purpose and usage of reporting events and its flow.
- How to format the list output.
- How to use the conditional and unconditional page breaks.
- Illustrate the Interactive reporting events.

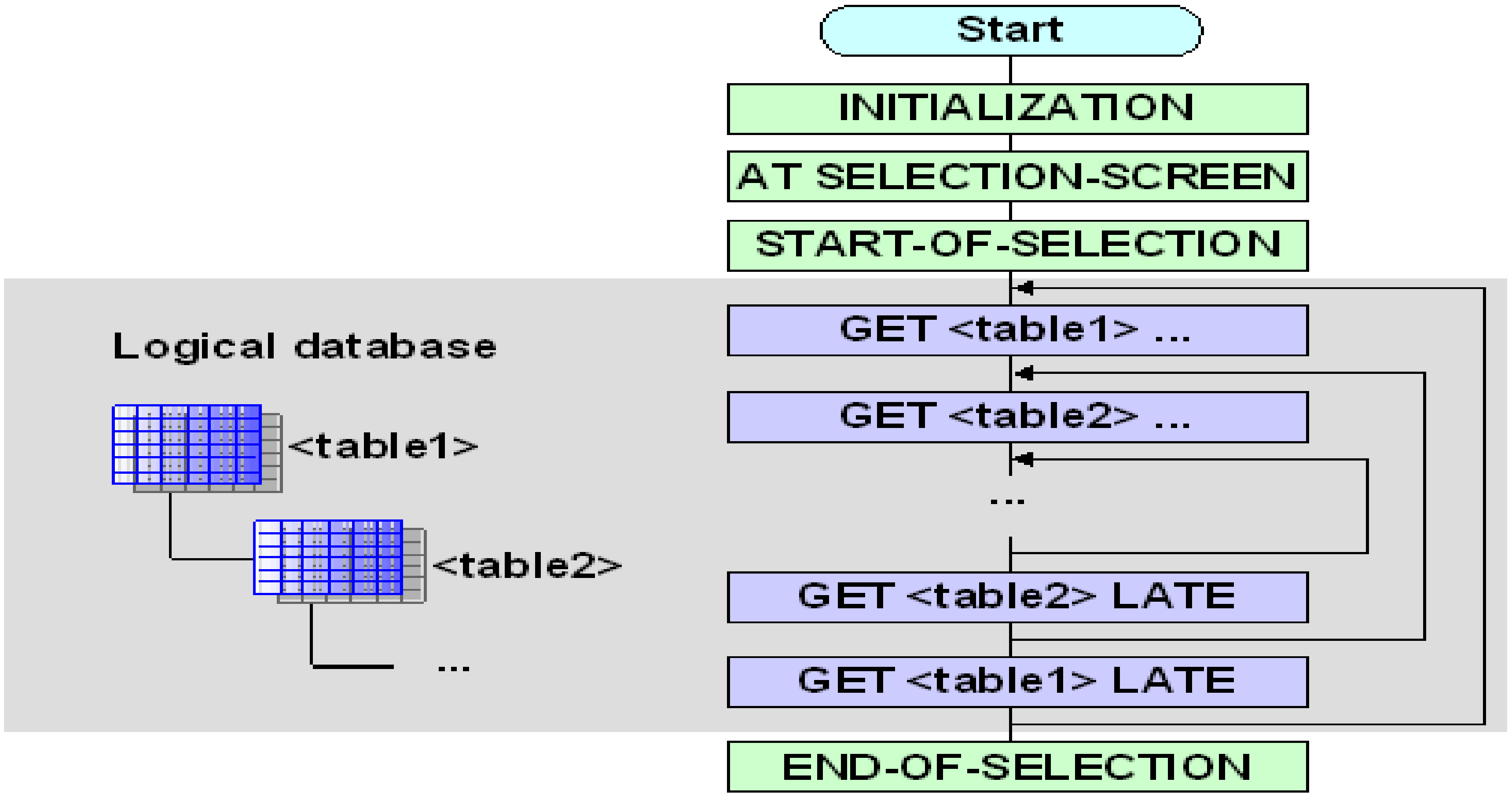
  AT LINE-SELECTION and AT USER-COMMAND.
- Illustrate using the menu painter.

Reporting Events

Event Blocks in Executable Programs

- When you run an executable program, the program flow is controlled by the external events in the ABAP runtime environment. The following diagram (Next Slide) shows the sequence of the events.
- The events in the gray box are only processed if you have entered a logical database in the program attributes. The AT SELECTION-SCREEN event is only processed if a selection screen is defined in the program or the logical database linked to the program.
- The other events occur when any executable program is runs.

Reporting Events



Event Blocks in Executable Programs

The following events occur when you run a typical executable program

- INITIALIZATION: Before the standard selection screen is displayed.
- AT SELECTION-SCREEN: After user input on a selection screen has been processed, but while the selection screen is still active.
- START-OF-SELECTION: After the standard selection screen has been processed, before data is read from the database.
- END-OF-SELECTION: After all data has been read by the logical database.
- TOP-OF-PAGE: In list processing when a new page starts.
- END-OF-PAGE: In list processing when a page ends.
- AT LINE-SELECTION: Event is triggered by either the user double clicking a particular line or using F2 to select it.
- AT PF<NN>: When the user triggers the predefined function code PF<NN>.
- AT USER-COMMAND: When the user triggers a function code defined in the program.
- TOP-OF-PAGE DURING LINE SELECTION: Event called during list processing when a detailed list is called.

Initialization

The input fields of the standard selection screen can only be initialized once the program has been started.

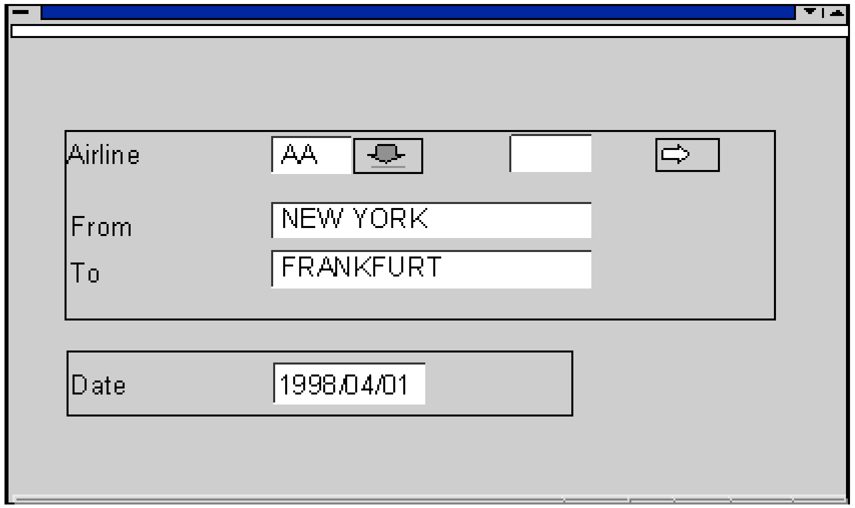
Processed before the presentation of the selection screen can be used to initialize values in the selection screen or to assign values to any parameters or the select-options that appear on the selection screen.

If an executable program declares a standard selection screen, the same program will be automatically called again by the system once the selection screen has been executed. This triggers the INITIALIZATION event again.

Initialization

```
REPORT  ZCAP_REPORT.
PARAMETERS DATUM TYPE SY-DATUM DEFAULT SY-DATUM.
INITIALIZATION.
CITY_FR = 'NEW YORK'. CITY_TO = 'FRANKFURT'.
CARRID-SIGN = 'I'. CARRID-OPTION = 'EQ'.
CARRID-LOW = 'AA'. APPEND CARRID.
DATUM+6(2) = '01'.
```

Output:



Airline	AA
From	NEW YORK
To	FRANKFURT
Date	1998/04/01

Demo: Initialization Event

At Selection-Screen

This is the basic form of events which occurs when the selection screen is being processed.

It is used to validate the information which is entered in the selection screen.

The standard selection screen is called automatically in the mid of the INITIALIZATION and START-OF-SELECTION events, either in an executable program or in the logical database which is linked to it.

Occurs after all the input data is passed to the underlying ABAP program from selection screen.

On passing the input data from Selection Screen to ABAP Program by the runtime environment, AT SELECTION-SCREEN event is triggered.

At Selection-Screen

Processing block is started after the user has specified all the criteria in the selection screen.

Selection Screen Processing:

- Started after the INITIALIZATION event triggered.
- Other events may be triggered for fields or for F4 help, depending upon user action on the selection screen.

This event can also be called on a particular parameter or select-option using At Selection-Screen on <parameter or select-option>.

To modify the Selection Screen elements before display, AT SELECTION-SCREEN OUTPUT event is used.

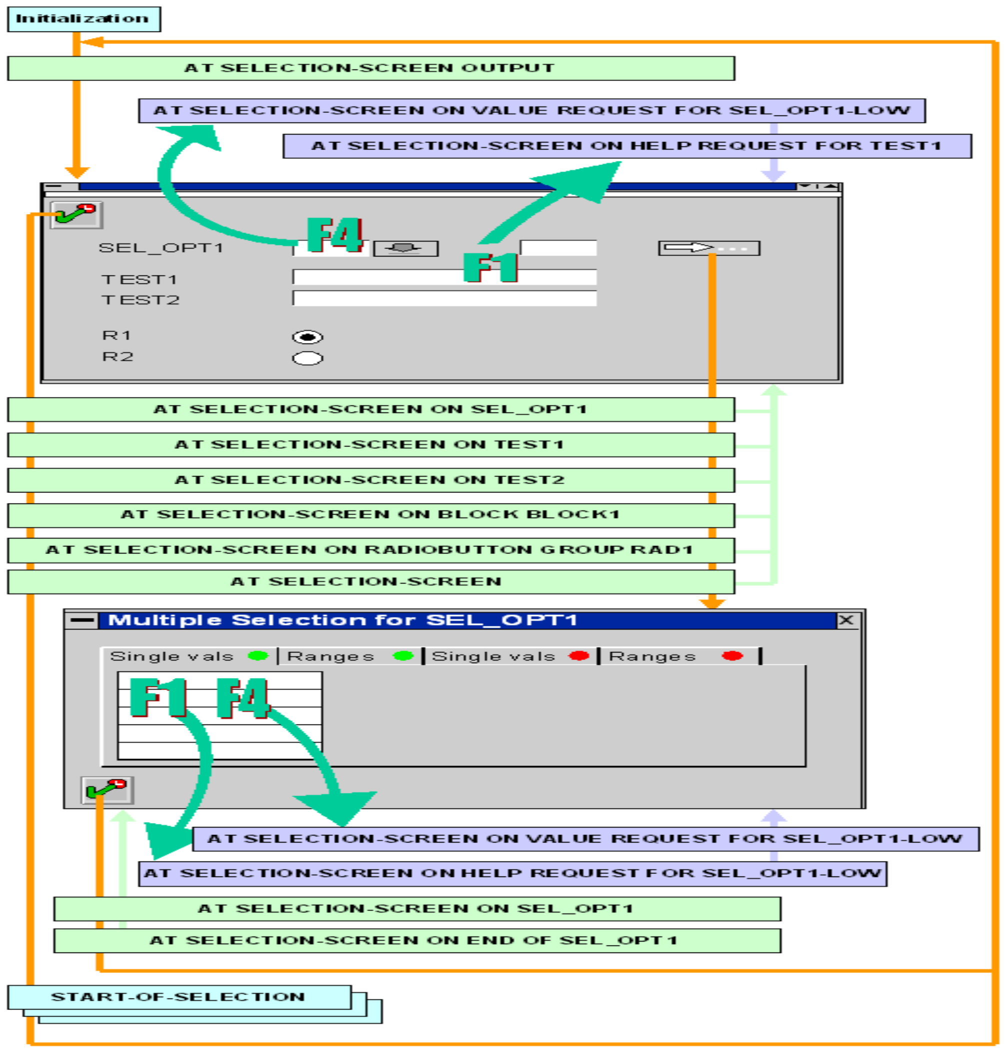
At Selection-Screen

```
DATA FIELD1(10).

SELECT-OPTIONS SEL_OPT1 FOR FIELD1.

SELECTION-SCREEN BEGIN OF BLOCK BLOCK1.
  PARAMETERS:   TEST1(10),
                TEST2(10).
SELECTION-SCREEN END OF BLOCK BLOCK1.

PARAMETERS: R1 RADIOBUTTON GROUP RAD1 DEFAULT 'X',
            R2 RADIOBUTTON GROUP RAD1.
```


Initialization
AT SELECTION-SCREEN OUTPUT
AT SELECTION-SCREEN ON VALUE REQUEST FOR SEL_OPT1-LOW
AT SELECTION-SCREEN ON HELP REQUEST FOR TEST1
F4
F1
SEL_OPT1
TEST1
TEST2
R1
R2
AT SELECTION-SCREEN ON SEL_OPT1
AT SELECTION-SCREEN ON TEST1
AT SELECTION-SCREEN ON TEST2
AT SELECTION-SCREEN ON BLOCK BLOCK1
AT SELECTION-SCREEN ON RADIOBUTTON GROUP RAD1
AT SELECTION-SCREEN
Multiple Selection for SEL_OPT1
Single vals
Ranges
Single vals
Ranges
F1
F4
AT SELECTION-SCREEN ON VALUE REQUEST FOR SEL_OPT1-LOW
AT SELECTION-SCREEN ON HELP REQUEST FOR SEL_OPT1-LOW
AT SELECTION-SCREEN ON SEL_OPT1
AT SELECTION-SCREEN ON END OF SEL_OPT1
START-OF-SELECTION

Demo: At Selection Screen Event

Start-of-Selection

The associated event is raised by the ABAP Runtime environment during an executable program.

It is used to prepare the required data for reading and also creating the list.

Processing block is executed after processing the selection screen

All the data is selected in this block.

All the main processing on the data except for interactive reporting is handled in this block.

Event Occurs at:

- After the selection screen has been processed.

Non-declarative statements which are written in between the report and first processing block are also processed in start of selection event.

End-of-Selection

This is the last event which is processed by the system.

Data which is selected and has been processed is printed to the screen in this block

This event is triggered after all the data has been read from the program/logical database before the list processor is being started.

It is also used to process the data that has been stored in sequential datasets like internal tables or extracts by the program

Exiting Event Blocks

Series of statements that allows to leave an event block in the program.

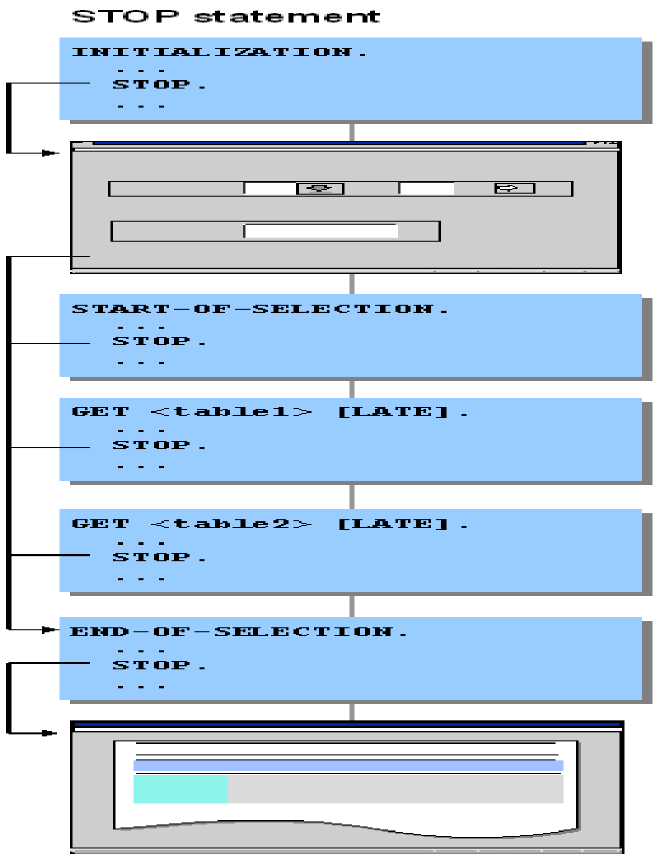
The processing of further statements depends on the statements and the events in which it is used.

Statements are used:

- STOP
- EXIT
- CHECK

Leaving Event Blocks using STOP

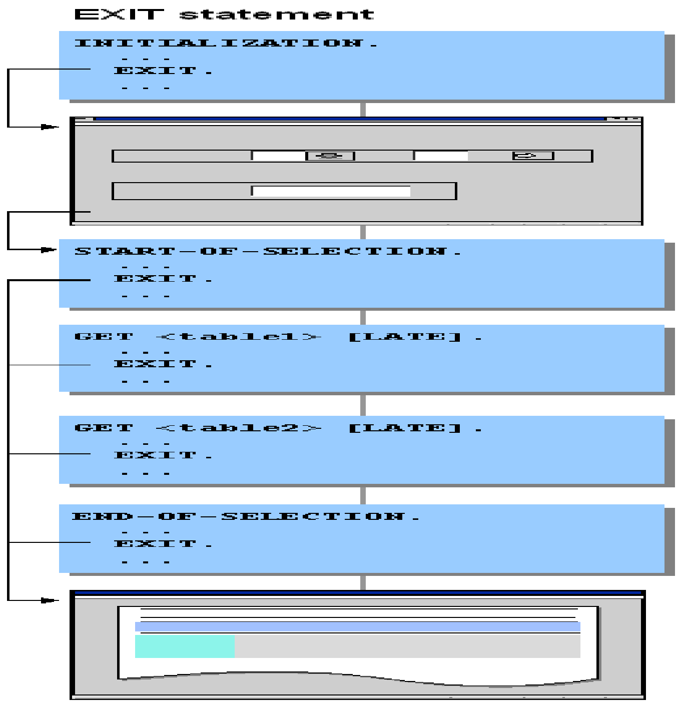
If you use the STOP statement within an event block, the system stops processing the block immediately. The ABAP runtime environment triggers the next event according to the following diagram:



Leaving Event Blocks using EXIT

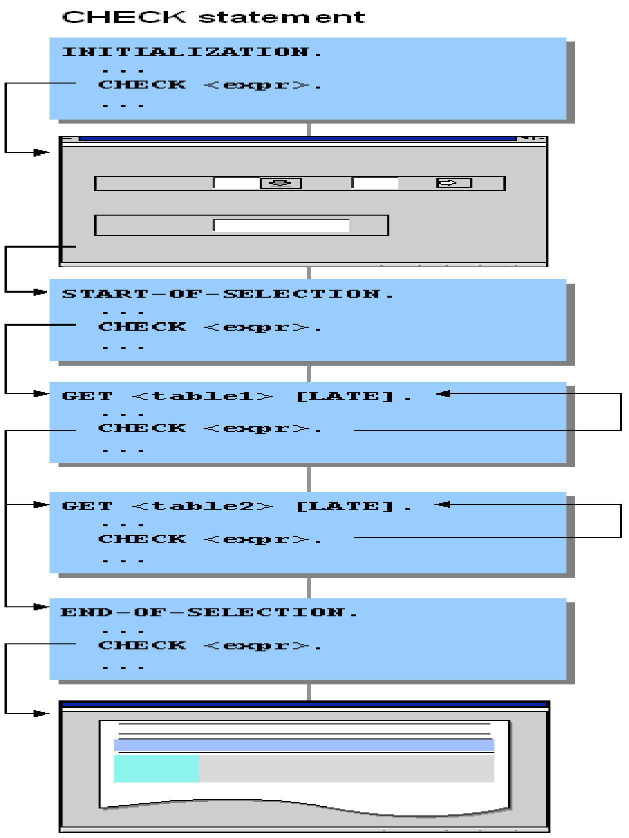
From the START-OF-SELECTION event onwards, the system starts the list processor directly when the EXIT statement occurs, and displays the list.

If the EXIT statement occurs in a loop using DO, WHILE, or LOOP, it is the loop that terminates, not the processing lock.



Leaving Event Blocks using CHECK

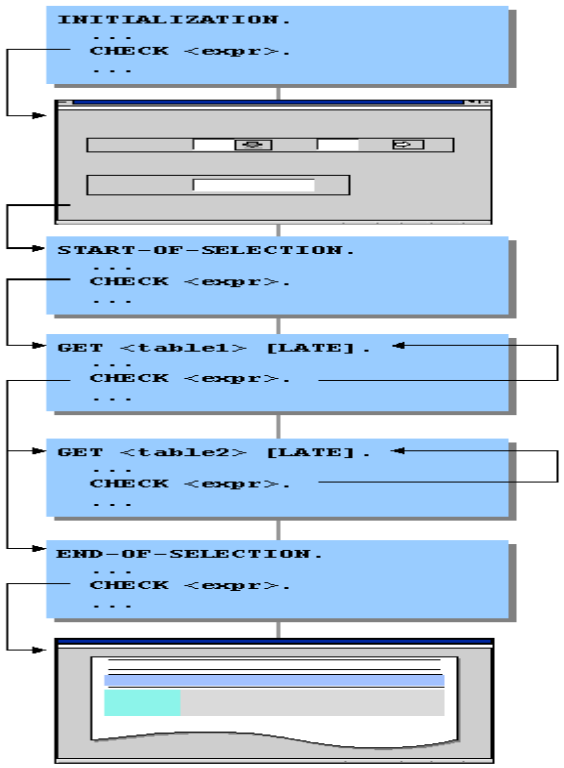
If you use the CHECK <EXPR> statement within an event block but not within a loop, and the condition <EXPR> is not fulfilled, the system exits the processing block immediately.



Leaving a GET event block using REJECT

The REJECT statement was specially developed for leaving GET event blocks. Unlike CHECK , EXIT and REJECT always refers to the current GET event block. The REJECT statement allows you to exit a GET event block directly from a loop or a subroutine.

CHECK statement

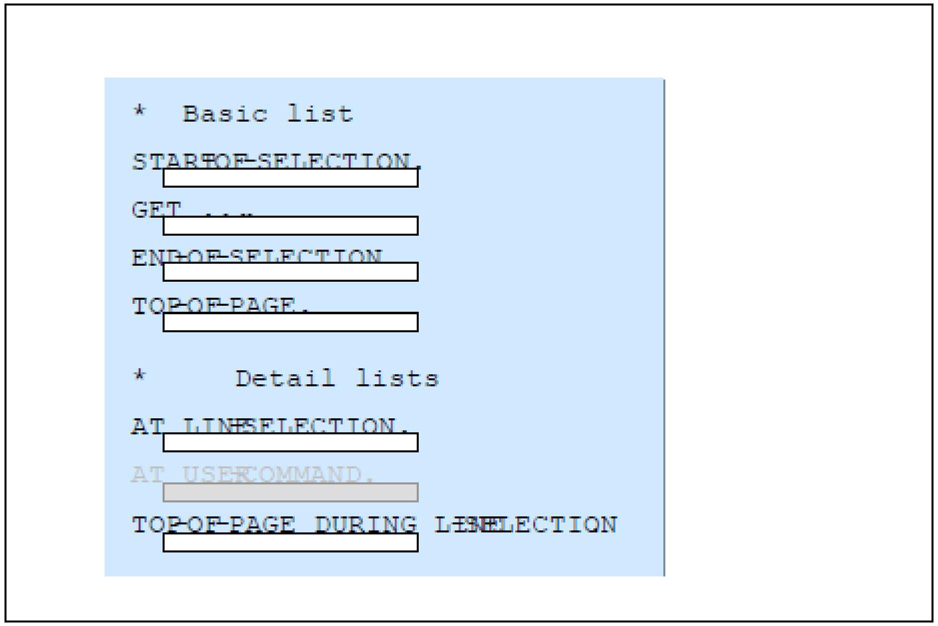


Event Blocks in Executable Programs

The events START-OF-SELECTION, GET, END-OF-SELECTION, TOP-OF-PAGE and END-OF-PAGE can be used only to create basic lists. Once you leave a basic list, these events are no longer processed.

Detail lists are created using two basic events: AT LINE-SELECTION AT PF Status and AT USER-COMMAND.

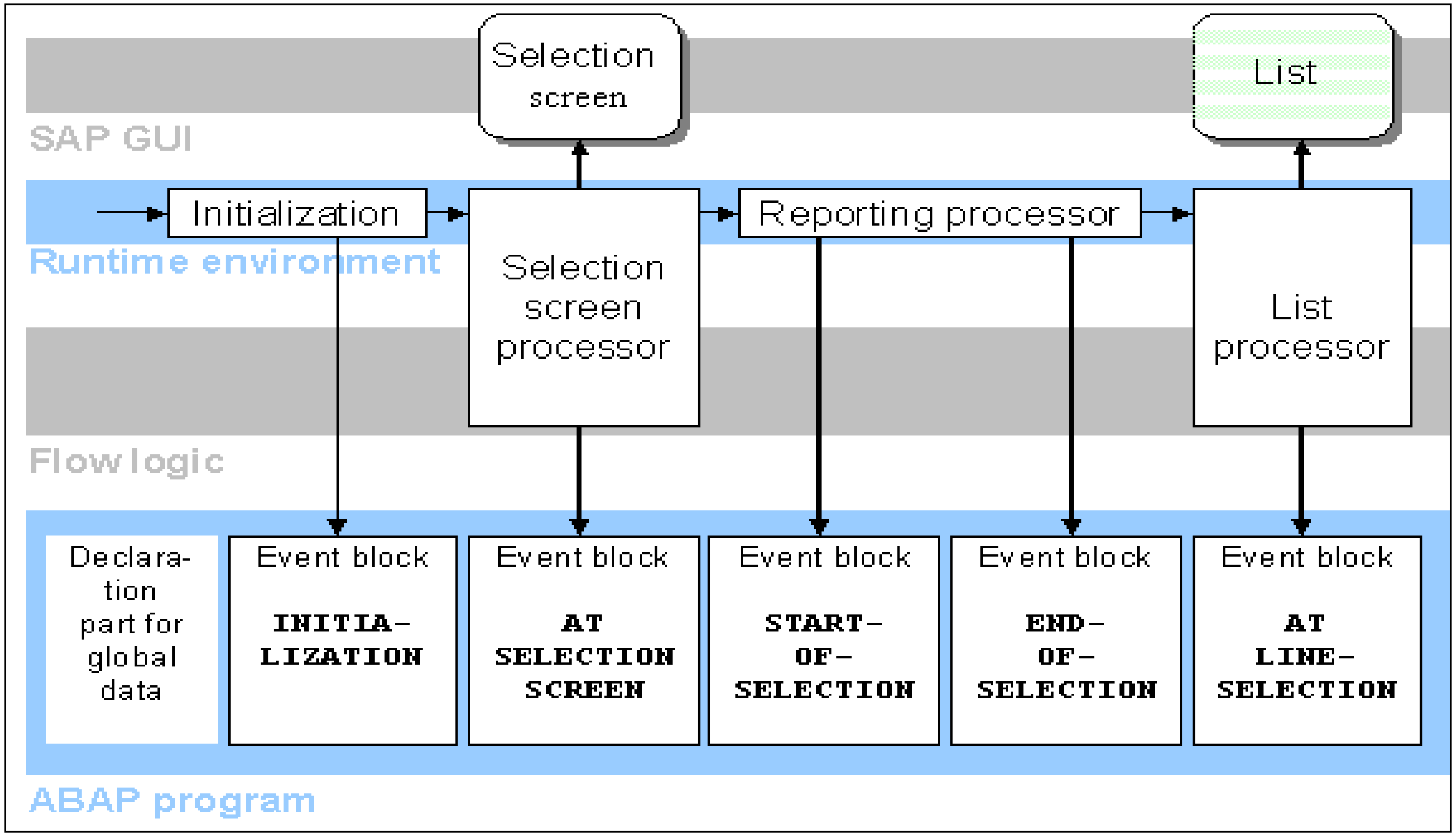
The event TOP-OF-PAGE DURING LINE-SELECTION is used to create headers for all detail lists.



```
*  Basic list
START-OF-SELECTION.
GET ....
END-OF-SELECTION.
TOP-OF-PAGE.

*     Detail lists
AT LINE-SELECTION.
AT USER-COMMAND.
TOP-OF-PAGE DURING LINE-SELECTION
```

Interactive Process Flow


Selection screen
List
SAP GUI
Initialization
Selection screen processor
Reporting processor
List processor
Runtime environment
Flow logic
Declaration part for global data
Event block INITIALIZATION
Event block AT SELECTION SCREEN
Event block START-OF-SELECTION
Event block END-OF-SELECTION
Event block AT LINE-SELECTION
ABAP program

Interactive Reporting Events

Lists are the output medium for structured, formatted data from ABAP programs.

In list processing the event will be intercepted by list processor and the list is processed.

One of the following list events may be called, that depends on the function code which is triggered by the user.

- AT LINE-SELECTION
- AT USER-COMMAND
- AT PF-STATUS

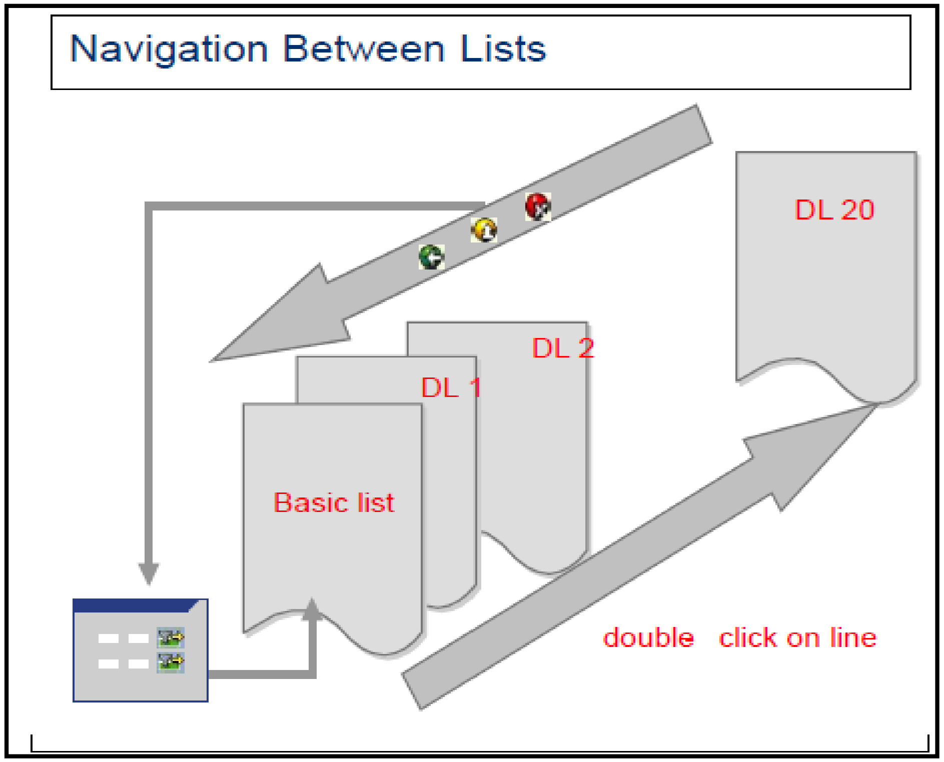
AT Line-Selection

If the user double-clicks on a report line, the AT LINE-SELECTION event is triggered. The event can also be triggered by single clicking on a report line and either pressing the Detail icon in the application tool bar, pressing F2, or entering the word PICK in the OK code field and pressing Enter.

A program can display up to 21 lists, out of which one is the basic list and 20 are the secondary lists. (Detailed List DL1, DL 2……and DL 20).

The basic list is nothing but the standard screen of an executable program.

Each list has in its own individual memory area called a list buffer



Creating Detail-Lists

Detail lists are the lists created during an interactive list event

Every time the user performs an action on a list, runtime environment will check whether there is an event block defined corresponding to the function code

If an event block exists, the system field SY-LSIND is automatically increased by one and the relevant event block is then executed

The list output that arises during this event block places its data into a new list (list level) with the index which is equal to SY-LSIND

By default, the new list overwrites the previous list, however the list can also be displayed in the form of dialog box

Creating Detail-Lists

Each interactive list event will create a new detail list

 If a list is created by the user on the next level , the system will store the previous list and displays the new one

The user can interact with whichever list is currently displayed

No standard page header can be defined for detail lists

Demo: Create Detail List using at Line Selection

Output Fields as Hotspots

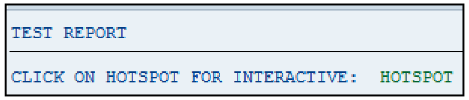
If the user clicks once onto a hotspot field, an event is triggered (for example, AT LINE-SELECTION).To output areas as hotspots, use the following option of the FORMAT statement:

Syntax: FORMAT HOTSPOT [ON|OFF].

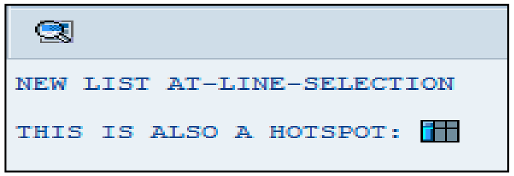
```
REPORT ZCAP_REP.
INCLUDE <LIST>.
START-OF-SELECTION.
  WRITE 'CLICK ON HOTSPOT FOR INTERACTIVE: '.
  FORMAT HOTSPOT ON COLOR 5 INVERSE ON. WRITE 'HOTSPOT'.
  FORMAT HOTSPOT OFF COLOR OFF.

AT LINE-SELECTION.
  WRITE / 'NEW LIST AT-LINE-SELECTION'. SKIP.
  WRITE 'THIS IS ALSO A HOTSPOT:'.
  WRITE ICON_LIST AS ICON HOTSPOT.
```

Output:



```
TEST REPORT
CLICK ON HOTSPOT FOR INTERACTIVE:  HOTSPOT
```



```
NEW LIST AT-LINE-SELECTION
THIS IS ALSO A HOTSPOT:
```

Consequences of Event Control

It is not possible to nest the Processing blocks

The event TOP-OF-PAGE cannot be used in secondary lists but TOP-OF-PAGE DURING LINE-SELECTION can be used

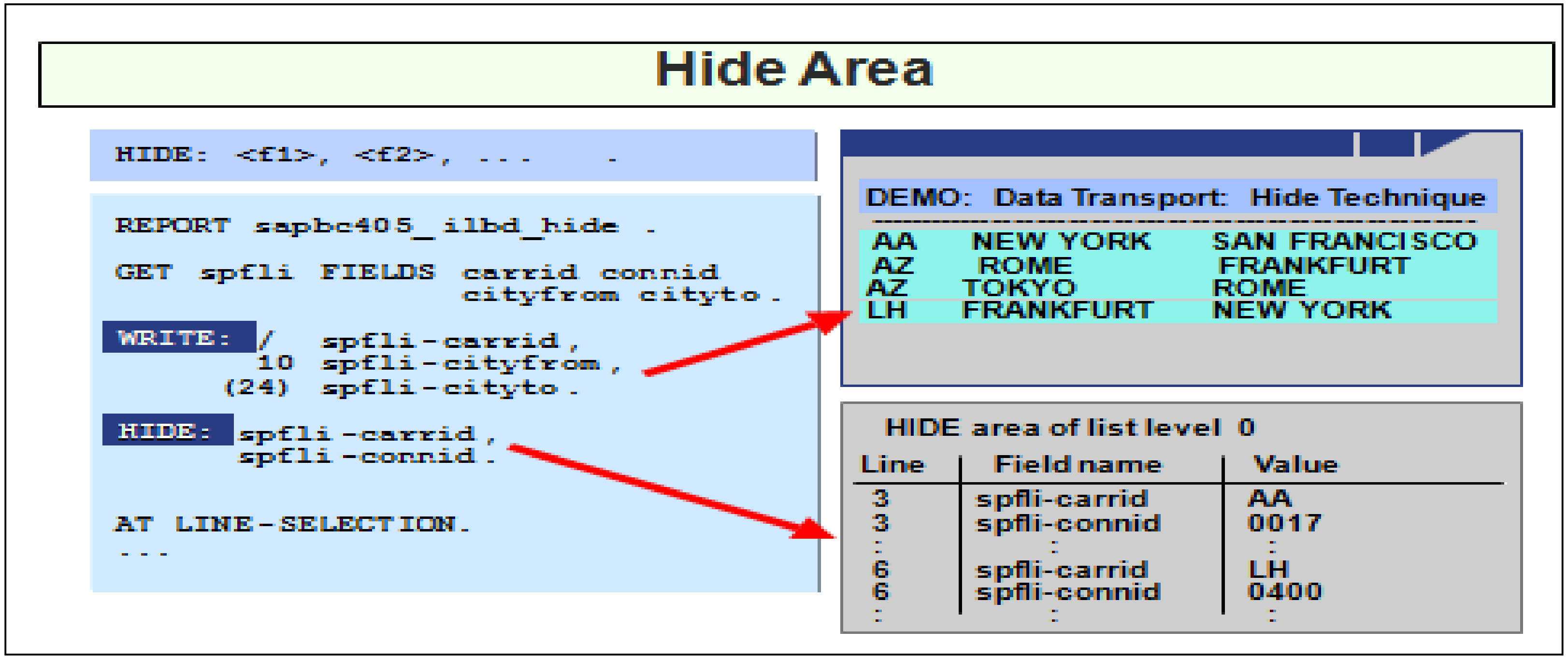
The event END-OF-PAGE in secondary lists is not processed by the system

The GET and GET LATE cannot be used to retrieve data for secondary lists

Data Transport by using HIDE

The HIDE keyword is used to store data objects and their values so they can be made available when the User selects a report line. When a line is selected, the fields that were hidden are filled with the values that you hid for that line.

It places the contents of the variable <f> for the current output line (system field SY-LINNO) into the HIDE Area. HIDE <f>.



```
HIDE: <f1>, <f2>, ...    .

REPORT sapbc405_ilbd_hide .

GET spfli FIELDS carrid connid
                 cityfrom cityto.

WRITE: / spfli-carrid,
      10 spfli-cityfrom,
    (24) spfli-cityto.

HIDE: spfli-carrid,
      spfli-connid.

AT LINE-SELECTION.
...
```

```
DEMO: Data Transport: Hide Technique
----------------------------------------------
AA    NEW YORK      SAN FRANCISCO
AZ     ROME          FRANKFURT
AZ    TOKYO         ROME
LH    FRANKFURT     NEW YORK
```

HIDE area of list level 0

Line	Field name	Value
3	spfli-carrid	AA
3	spfli-connid	0017
:	:	:
6	spfli-carrid	LH
6	spfli-connid	0400
:	:	:

Line Selection – Hide Statement

The user selects a line for which data has been stored in the HIDE area.

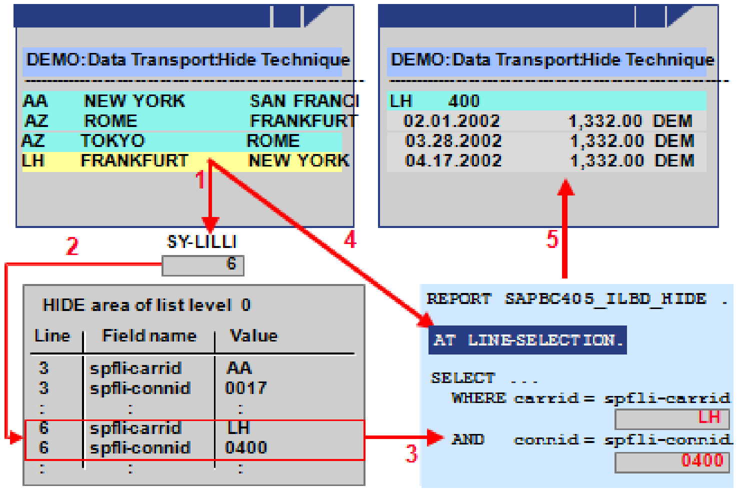
The runtime system evaluates field SY-LILLI to determine the selected line.

The runtime system jumps to the point in the HIDE area where data for this line is stored.

The runtime system then inserts all values stored for the selected line in the HIDE area into their corresponding fields.

The runtime system processes the event AT LINE-SELECTION and its corresponding program processing block.

A detail list is created.



```
DEMO:Data Transport:Hide Technique
----------------------------------------------
AA    NEW YORK      SAN FRANCI
AZ    ROME          FRANKFURT
AZ    TOKYO         ROME
LH    FRANKFURT     NEW YORK
```

```
DEMO:Data Transport:Hide Technique
----------------------------------------------
LH    400
  02.01.2002       1,332.00  DEM
  03.28.2002       1,332.00  DEM
  04.17.2002       1,332.00  DEM
```

HIDE area of list level 0

Line	Field name	Value
3	spfli-carrid	AA
3	spfli-connid	0017
:	:	:
6	spfli-carrid	LH
6	spfli-connid	0400
:	:	:

```
REPORT  SAPBC405_ILBD_HIDE .

AT  LINE-SELECTION.

SELECT  ...
  WHERE carrid = spfli-carrid
                          LH
  AND    connid = spfli-connid
                        0400
```

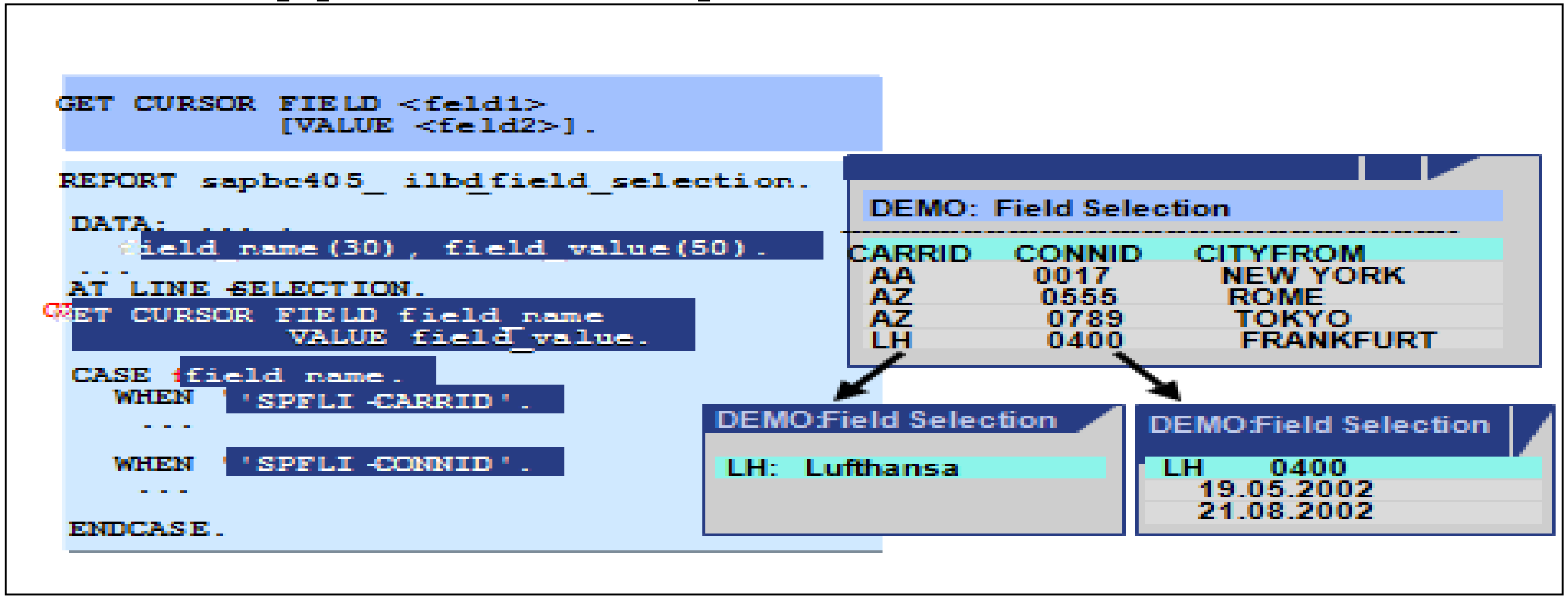
Demo: Hide Statement

Field Selection - GET CURSOR

GET CURSOR statement is used to create detail lists according to the cursor position.

The parameter FIELD provides the name of an output field. The parameter VALUE provide the output value.

GET CURSOR FIELD <f> [OFFSET <off>] [LINE <lin>] [VALUE <val>] [LENGTH <len>].



```
GET CURSOR FIELD <feld1>
              [VALUE <feld2>].

REPORT sapbc405_ ilbd field_selection.
 DATA: ...
    field_name(30), field_value(50).
 ...
 AT LINE SELECTION.
 GET CURSOR FIELD field_name
              VALUE field_value.
 CASE field_name.
   WHEN 'SPFLI-CARRID'.
    ...
   WHEN 'SPFLI-CONNID'.
    ...
 ENDCASE.
```

```
DEMO: Field Selection
CARRID   CONNID   CITYFROM
 AA       0017     NEW YORK
 AZ       0555     ROME
 AZ       0789     TOKYO
 LH       0400     FRANKFURT
```

```
DEMO:Field Selection
LH:  Lufthansa
```

```
DEMO:Field Selection
LH      0400
 19.05.2002
 21.08.2002
```

Demo: Get Cursor

Lists in Dialog Box

List can be displayed in a dialog box instead of on the full screen using the following WINDOW statement

```
WINDOW STARTING AT <left> <upper> [ENDING AT <right> <lower>].
```

The WINDOW statement takes effect only within the interactive event’s processing block which is only for detail lists

Dialog boxes do not have either menu bar or standard toolbar

The application toolbar is seen at the bottom of the dialog box, this feature is common to all dialog boxes in the R/3 System

Reading and Modifying List Line

READ - Reading a Line from a List

- READS LINE NUMBER LINE OF THE LIST, USUALLY AFTER A LINE SELECTION
  - READ LINE LINE.
  - READ LINE LINE OF CURRENT PAGE.
  - READ LINE LINE OF PAGE PAG
  - READ CURRENT LINE

MODIFY LINE n.

- Change a List Line
  - INDEX IDX - Changes the corresponding line in the list at list level IDX
  - FIELD VALUE f1 FROM G1 ... FN FROM GN

WINDOW STARTING AT X1 Y1 ENDING AT X2 Y2.

- Displays the current secondary list as a modal dialog box only up to 20 windows

Reading Lines from the Lists

Used to read a line from a list after an interactive list event.

```
READ LINE <lin> [INDEX <idx>] [FIELD VALUE <f1> [INTO <g 1>] ... <f n>
    [INTO <g n>]]  [OF CURRENT PAGE|OF PAGE <p>].
```

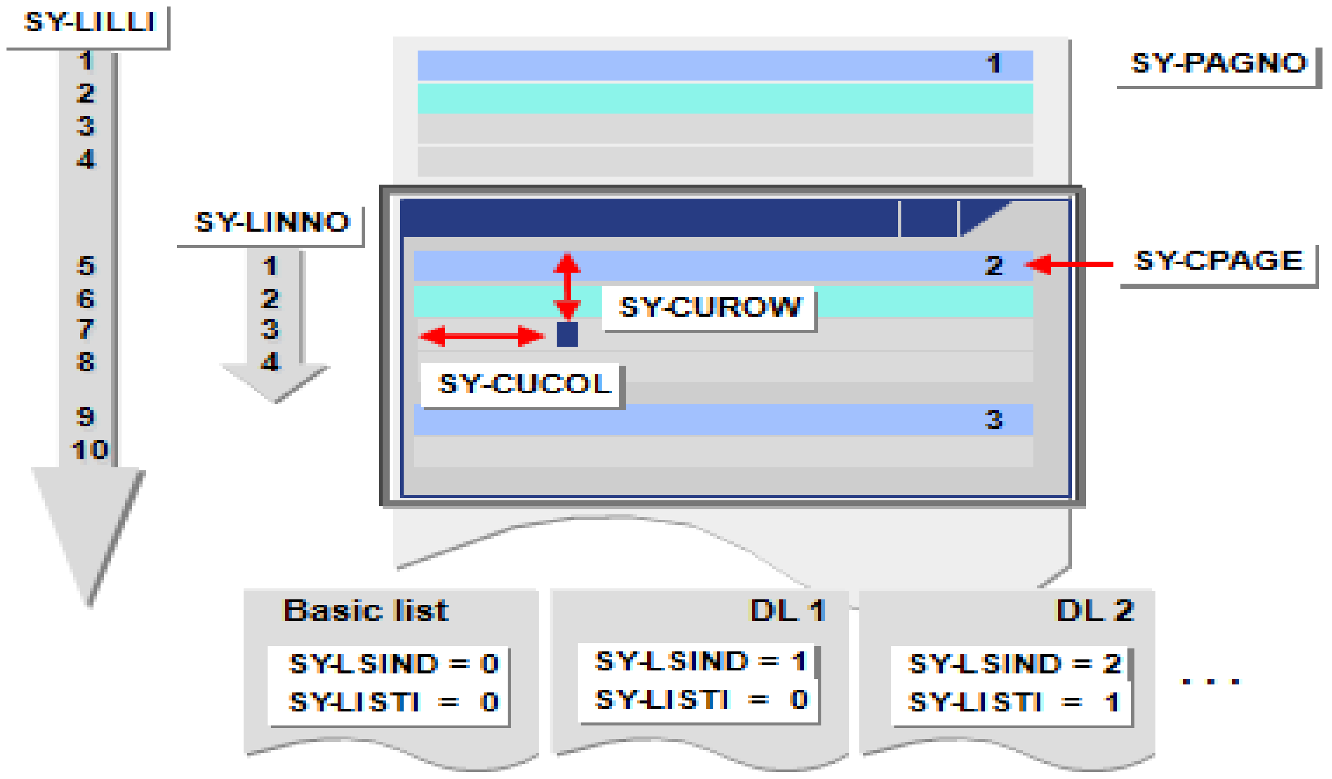
- The statement stores the contents of line <lin> from the list on which the event was triggered (index SY-LILLI) in the SY-LISEL system field and will fill all HIDE information that is stored for this line back to the corresponding fields

```
READ CURRENT LINE [FIELD VALUE <f1> [INTO <g 1>] ...].
```

- This statement reads a line read before by an interactive event or by READ LINE

Demo: Read Lines from List

System Fields for Interactive Lists



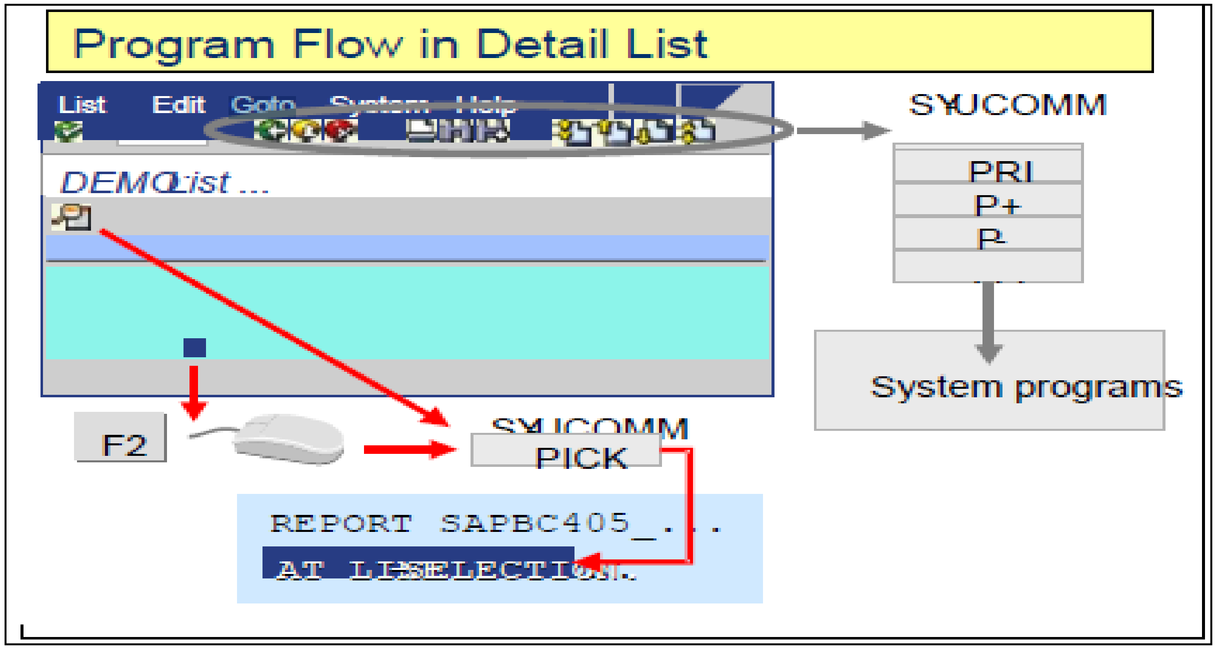
System Fields for Interactive Lists

SY-LSIND	Index for the current list.
SY-LISTI	Index of the last list displayed.
SY-LILLI	Absolute number of a selected line in the list displayed.
SY-CPAGE	Number of the upper-most displayed line in the list displayed.
SY-LISEL	Contents of the line from which the event was triggered.
SY-CUCOL	Number of the column in the window where the cursor was last positioned in the list displayed.
SY-CUROW	Number of the line in the window where the cursor was last positioned in the list displayed.
SY-STACO	Number of the first column displayed in the list displayed.
SY-STARO	Number of the first visible line in the top displayed page (SY- PAGE) in the list displayed (not including header lines).
10.SY-UCOMM	Function code that triggered the interactive event in the list displayed.
11.SY-PFKEY	Status of the list displayed.

AT USER-COMMAND

If the report has a custom GUI and the user selects a menu option, the event AT USER-COMMAND is triggered. The function code assigned to the menu option can be evaluated using a CASE construct to determine which menu option was selected.

All other function codes are either intercepted by the runtime environment or trigger the event AT USER-COMMAND.



AT USER-COMMAND

Function codes that trigger AT USER-COMMAND must be defined in the own GUI status.

GUI status for lists can be defined and attached to list level using

- SET PF-STATUS in executable program.

Function codes are used for this purpose and they will be maintained in the GUI status of the list screen.

The GUI status will be maintained in Menu Painter tool in the ABAP Workbench.

The system assigns function codes to user actions that are list-specific.

Dialog Status and Title in program

- Setting a Dialog-Status

```
SET PF-STATUS <stat> [EXCLUDING <f>|<itab>]
                     [OF PROGRAM <prog>].
```

- This statement sets the status <stat> for the current output list.
- SET PF-STATUS SPACE is used to set the standard list status.

Setting a Title for a List

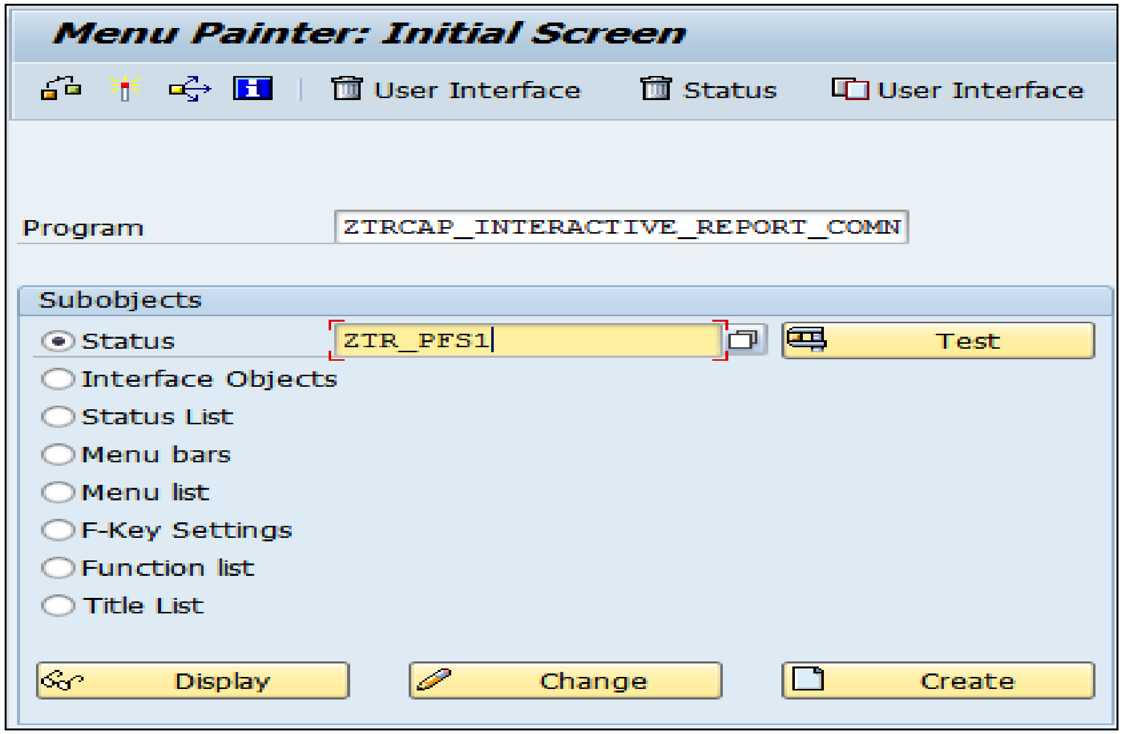
```
SET TITLEBAR <ttl> [WITH <g1> ... <g9>]
               [OF PROGRAM <prog>].
```

- The above statement will set the title of the user interface for the output list.
- This will remain active for all screens until another SET TITLEBAR is specified.

Menu Painter

- It is a tool used to design the user interfaces for ABAP Programs.
- The GUI status will be maintained in Menu Painter tool in the ABAP Workbench
- GUI status means, the instance of the interface consisting of menu bar, a standard toolbar, an application toolbar, and a function key setting.

- The GUI status and GUI
  - title defines how the user
  - interface will look and
  - behave in an ABAP program.
  - T-CODE SE41.



Program ZTRCAP_INTERACTIVE_REPORT_COMN

Subobjects

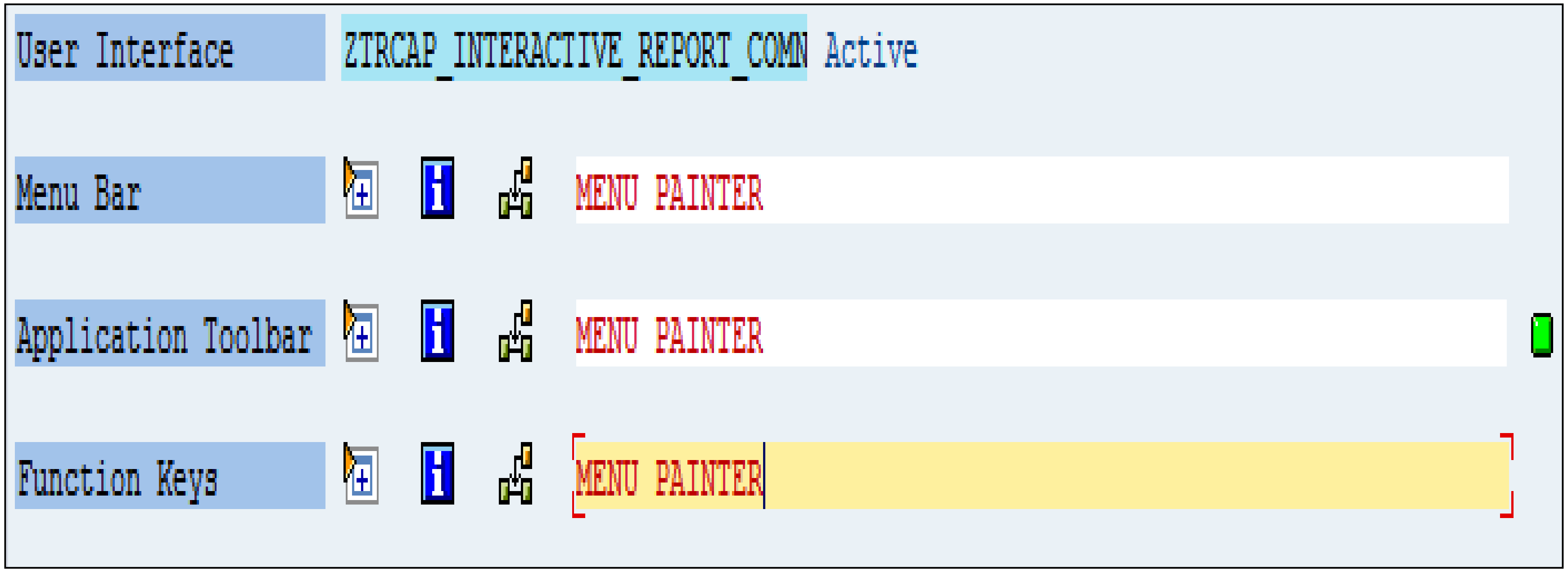
[✓ Status] ZTR_PFS1

Interface Objects
Status List
Menu bars
Menu list
F-Key Settings
Function list
Title List

Menu Painter

Menu Bar Consists of the following the options.

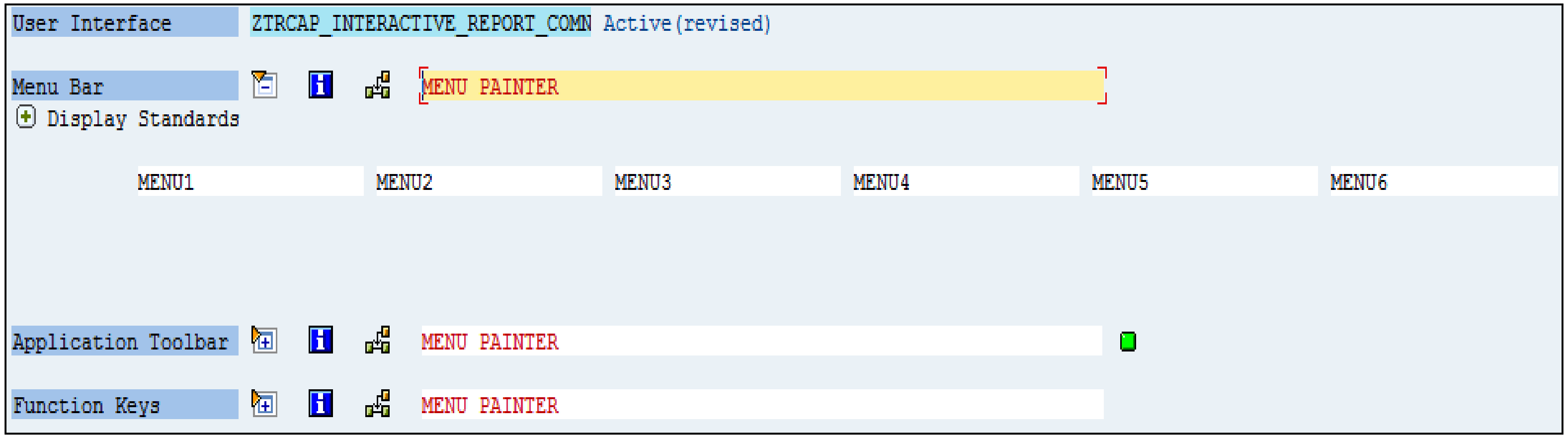
- Menu Bar
- Application Toolbar
- Function Keys



User Interface	ZTRCAP_INTERACTIVE_REPORT_COMN	Active
Menu Bar	MENU PAINTER	
Application Toolbar	MENU PAINTER	
Function Keys	MENU PAINTER	

Menu Painter - Menu Bar

- A menu bar may contain six menus.
- Two standard menus are added by default
  - System
  - Help
- The two menus are displayed at the right-hand end of the menu bar



User Interface ZTRCAP_INTERACTIVE_REPORT_COMN Active(revised)

Menu Bar MENU PAINTER

Display Standards

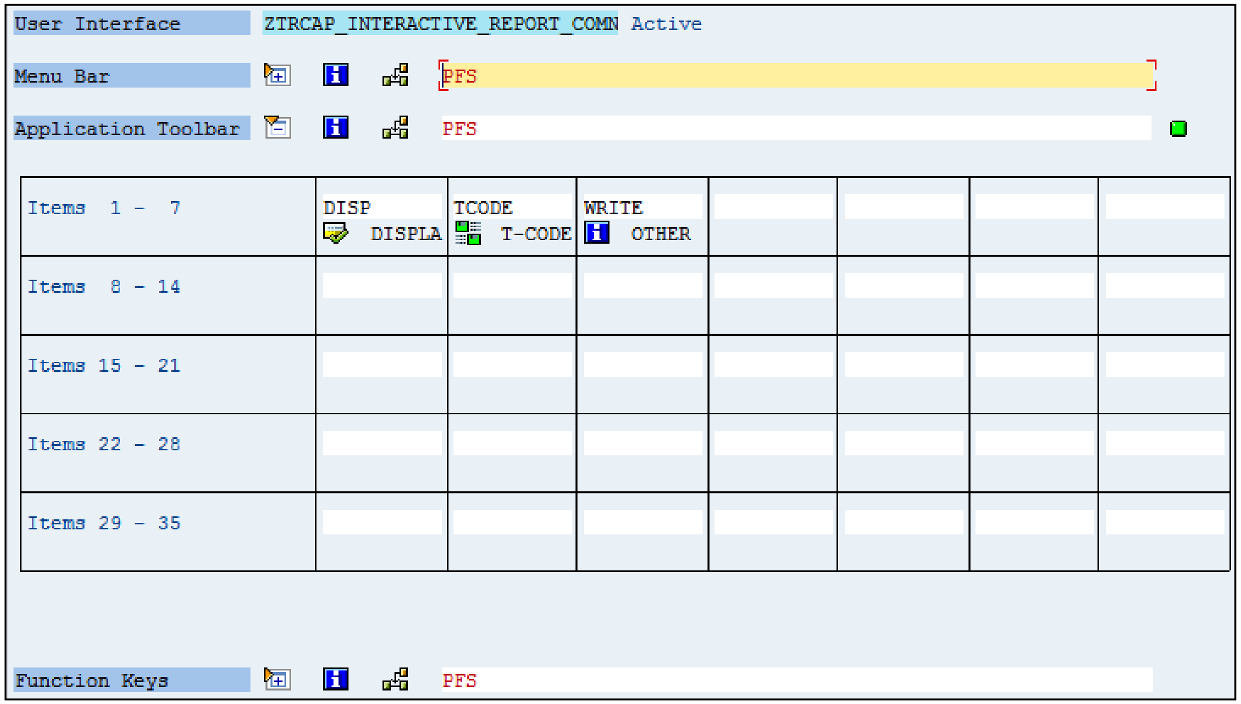
MENU1	MENU2	MENU3	MENU4	MENU5	MENU6

Application Toolbar MENU PAINTER

Function Keys MENU PAINTER

Menu Painter - Application Toolbar

- A function to a function key must be assigned before it can be included in the application toolbar.
- A maximum of 35 pushbuttons in the application toolbar can be included.
- Application toolbar functions contains
  - Icon
  - Text or
  - Both Icon and Text



User Interface ZTRCAP_INTERACTIVE_REPORT_COMN Active

Menu Bar PFS

Application Toolbar PFS

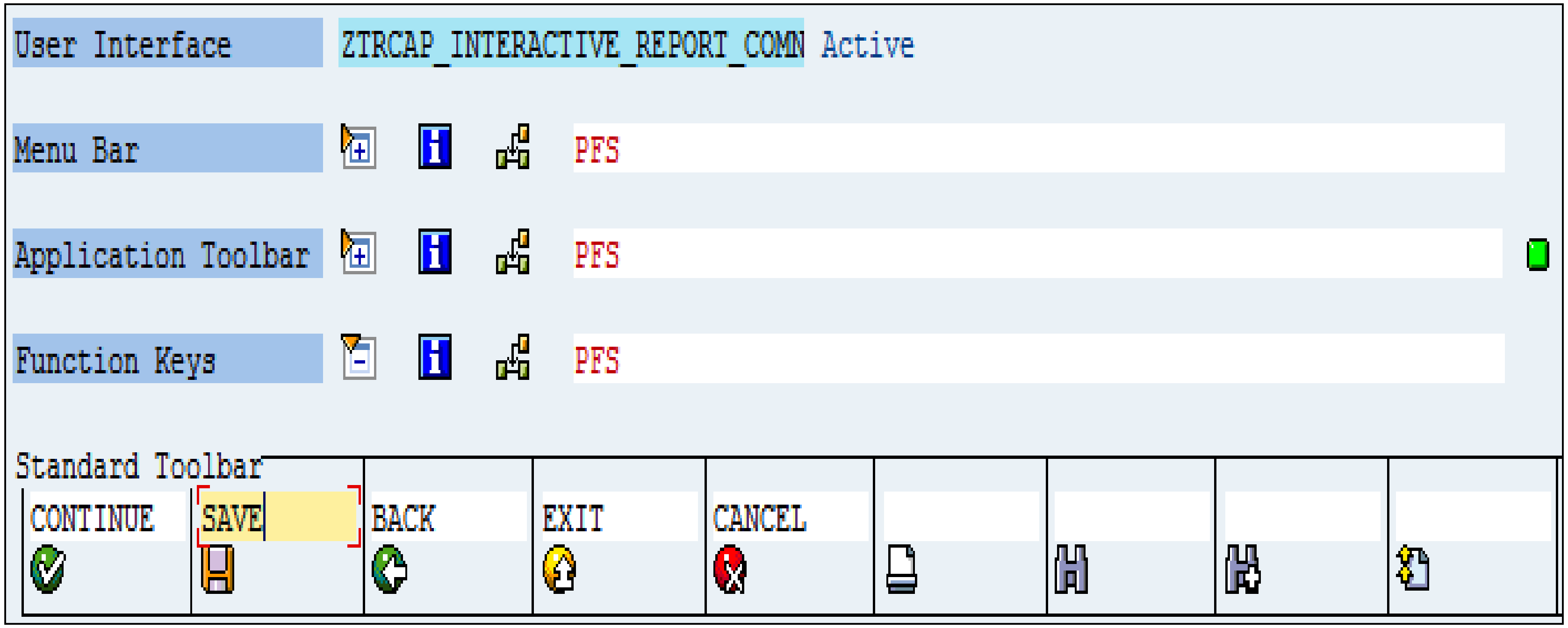
Items 1 - 7	DISP DISPLA	TCODE T-CODE	WRITE OTHER				
Items 8 - 14							
Items 15 - 21							
Items 22 - 28							
Items 29 - 35							

Function Keys PFS

Menu Painter - Function Keys

A function keys contain the.

- Standard Toolbar.
- Recommended Function Key Settings.
- Freely Assigned Function Keys.



User Interface	ZTRCAP_INTERACTIVE_REPORT_COMN Active
Menu Bar	PFS
Application Toolbar	PFS
Function Keys	PFS

Standard Toolbar

CONTINUE	SAVE	BACK	EXIT	CANCEL				

Review Question

Question 1. _______event occurs before the standard selection screen is called.

Question 2: The _____ keyword is used to store data objects.

Question 3: ______ is a tool used to design the user interfaces for ABAP Programs.

Summary

In this lesson, you have learnt:

- To Generate a list using simple reports
- To use different events in the report
- To Illustrate events triggering interactive reports
- To Develop programs on Interactive reports by using the
  At Line-Selection and At User-Command Events.
- To Design Menus, Function Keys and application tool bar.