

# List Box Handling

This section describes how to maintain two list boxes and the communication between them. The first list box displays a set of objects from which entries can be selected and transferred to the second list box.

- **List box ALL**  
Contains all available objects.  
Example: All articles required for an order.
- **List box SELECTED**  
Contains all objects selected from list box ALL.  
Example: All articles selected to create an invoice.

The following topics are covered below:

- Prerequisites
  - Functional Scope of the Frame Modules
  - Additional User Activities
  - Integrate Processing into the Dialog
- 

## Prerequisites

The following modules for the frame modules are prerequisites for list box processing:

- **ZXFXLB0C** - This copycode contains the general code for the integration of list box processing in a dialog.
- **ZXFXLB1C** - This copycode contains the special-purpose code for the integration of list box processing in a dialog.
- **ZXFXLB2C** - This copycode contains the general code for the integration of list box processing in a subprogram.
- **ZXFXLB0N** - This subprogram is the skeleton for the creation of list box processing.
- **ZXFXLB0L** - This local data area contains all variables required by the subprograms for list box processing.
- **ZXFXLB0A** - Standard parameter data area for list box processing for the subprogram.
- **ZXFXLB01A** - Standard parameter data area for list box processing for list box ALL. In addition, this parameter data area is the basis for the creation of individual interfaces.
- **ZXFXLB02A** - Standard parameter data area for list box processing for list box SELECTED. In addition, this parameter data area is the basis for the creation of individual interfaces.

In addition, the access modules and their parameter areas are required to fill and process both list boxes. Multiple-record access module and the parameter for the object type whose occurrences are displayed in the list box. For example: Item.

Single-record access module and the parameter for the object type for which selections are carried out. For example: Invoice.

## Functional Scope of the Frame Modules

The following functionality is provided by the frame modules:

- Search using a Start Value in List Box ALL.  
For the list box ALL, which may have a large number of entries, a start value can be provided. When the Search button is selected, the list box is displayed beginning with the specified start value.

- Dynamic read for list box ALL.  
To minimize the response time for large data volumes, list box ALL is at first not filled with all possible entries. Instead, the data is dynamically read as required when the user pages downwards within the list.
- Analysis of the Selection in list Box ALL.  
The following is checked whenever a user selects one or more entries in the list box ALL:
  - Maximal number selected?  
Does list box SELECTED already contain the maximum permitted entries?  
If it does, no further selection is possible and the Select button is not activated.  
Example: The frame guarantees that no more than 50 items can be selected for one invoice.
  - Double selection?  
Does list box SELECTED already contain the entries selected in list box ALL?  
If it does, repeated selection is not possible and the Select button is not activated.  
Example: The frame guarantees that each item can be selected only once per invoice.
  - Selection  
All selected entries in list box ALL are transferred to list box SELECTED - if not previously done - by using the Select button or a double-click on the list box entry.

The following options are available when adding new entries to the list box SELECTED:

- Insert in sort sequence.  
New entries are inserted in the sorting sequence if the list box was implemented as a sorted list box (Mark attribute SORTED in the dialog editor).
- Insert at end of list.  
New entries are added at the end of list, when the list box #LB\_SELECTED is not implemented as a sorted list box (attribute SORTED in the dialog editor is not marked) and the user has not selected an entry in this list box.
- Insert at a specific position within a list.  
New entries are inserted at a specific position, when the list box SELECTED is not implemented as a sorted list box (attribute SORTED in the dialog editor is not marked) and the user has selected an entry in this list box. In this case, the new entry is inserted before the selected entry.
- Remove.  
The selected entries in list box SELECTED are removed when the Remove button is selected.

## Additional User Activities

### Subprogram

Copy the subprogram ZXFXMLB0N to subprogram xxxXMLB0N and adapt it as described in the suggested code.

Copy the parameter data area ZXFXMLB1A to parameter data area xxxXMLB1A.

Copy the parameter data area ZXFXMLB2A to parameter data area xxxXMLB2A.

### Copycode

Copy copycode ZXFXMLB1C to copycode xxxXMLB1C and adapt it as described in the suggested code. Thus the inline subroutine PROCESS\_LISTBOX\_xx\_yy is created.

### Dialog Layout

Create the following graphical elements for a dialog:

#### List Box ALL

Name HANDLE according to copycode xxxXMLB1C.

Mark the attribute MULTIPLE SELECTION if several entries are to be selected at one time.

Call subroutine PROCESS\_LISTBOX\_xx\_yy in the event handlers for CLICK, DOUBLE-CLICK and FILL.

#### List box SELECTED

Name HANDLE according to copycode xxxXMLB1C.

Mark the attribute MULTIPLE SELECTION if several entries are to be selected at one time.

Call subroutine PROCESS\_LISTBOX\_xx\_yy in the event handlers for CLICK and DOUBLE-CLICK.

#### Search Push button

Assign COMMAND-ID according to copycode xxxXMLB1C.

#### Add Push button

Assign COMMAND-ID according to copycode xxxXMLB1C.

#### Delete Push button

Assign COMMAND-ID according to copycode xxxXMLB1C.

## Assign Data Areas

In addition, assign the following global data areas and parameter data areas:

- ZXFXMLB0A
- xxxXMLB1A
- xxxXMLB2A

## Include Copycode

Place copycode xxxXMLB1C in the AFTER ANY event of the dialog.

# Integrate Processing into the Dialog

## Subroutine Z\_INITIALIZE

Add the following line:  
INCLUDE ZXFXMLB4C

## Subroutine Z\_FILL\_DIALOG

Add the following lines:  
INCLUDE ZXFXMLB3C  
PERFORM PROCESS\_LISTBOX\_xx\_yy

## Subroutine Z\_CUSTOM\_CMD

Call subroutine PROCESS\_LISTBOX\_xx\_yy for commands *Search*, *Add*, and *Delete*

## Subroutine Z\_UPDATE\_PREC\_REC

To re-transfer data from list box SELECTED to the respective access module parameter data area, add the following lines at the respective position:

```
MOVE LZ_LB_RETURN TO PZ_LISTBOX.PZ_OPERATION  
  
PERFORM PROCESS_LISTBOX_xx_yy
```