Working with List Box Controls and Selection Box Controls

Working with List Box Controls and Selection Box Controls

List box controls and selection box controls contain a number of items. Both the controls and the items are dialog elements; the controls are the parents of the items.

There are two ways of creating list box items and selection box items:

- Use Natural code to create individual and multiple list box items dynamically; or
- use the dialog editor (to add single or arrays of list box items and selection box items).

In Natural code, this may look like this:

```
#AMOUNT := 5
ITEM (1) := 'BERLIN'
ITEM (2) := 'PARIS'
ITEM (3) := 'LONDON'
ITEM (4) := 'MILAN'
ITEM (5) := 'MADRID'
PROCESS GUI ACTION ADD-ITEMS WITH #LB-1 #AMOUNT #ITEM (1:5) GIVING #RESPONSE
```

You first specify the number of items you want to create, name the items, and use the `PROCESS GUI` statement action `ADD-ITEMS`.

If you want to go through all items of a list box control to find out which ones are selected, it is advisable to use the `SELECTED-SUCCESSOR` attribute because if a list box control contains a large number of items (100, for example), this helps improve performance. If you use `SELECTED-SUCCESSOR`, you have one query instead of 100 individual queries if you use the attributes `SELECTED` and `SUCCESSOR`.

Example:

```
/* Displays the STRING attribute of every SELECTED list-box item
MOVE #LISTBOX.SELECTED-SUCCESSOR TO #LBITEM
REPEAT UNTIL #LBITEM = NULL-HANDLE
   .../* STRING display logic

    MOVE #LBITEM.SELECTED-SUCCESSOR TO #LBITEM
END-REPEAT
```

For performance reasons, you should not use the `SELECTED-SUCCESSOR` attribute to refer to the same dialog element handle twice, because Natural goes through the list of item handles twice:

```
/* Displays the STRING attribute of every SELECTED list-box item,
/* but may be slow
MOVE #LISTBOX.SELECTED-SUCCESSOR TO #LBITEM
REPEAT UNTIL #LBITEM = NULL-HANDLE
   IF #LBITEM.SELECTED-SUCCESSOR = NULL-HANDLE /* Searches in the list of items
     IGNORE
   END-IF
   .../* STRING display logic
   MOVE #LBITEM.SELECTED-SUCCESSOR TO #LBITEM /* Searches in the list of items
END-REPEAT                                    /* for the second time
```

Working with List Box Controls and Selection Box Controls

To avoid this problem, you use a second variable #OLDITEM besides #LBITEM:

```
/* Displays the STRING attribute of every SELECTED list-box item
MOVE #LISTBOX.SELECTED-SUCCESSOR TO #LBITEM
REPEAT UNTIL #LBITEM = NULL-HANDLE
   #OLDITEM = #LBITEM
   #LBITEM = #LBITEM.SELECTED-SUCCESSOR/* Searches in the list of items (once)
   IF #LBITEM = NULL-HANDLE
     IGNORE
  END-IF
   .../* Display logic using #OLDITEM.STRING
END-REPEAT
```

If you retrieve the handle values of the selected items, a value other than NULL-HANDLE would normally be returned by selected items. Such a handle value can also be returned by non-selected items if you assign SELECTED-SUCCESSOR a value immediately before retrieving the SELECTED-SUCCESSOR value of a non-selected item, as shown in the following example:

```
...
PTR := #LB-1.SELECTED-SUCCESSOR
PTR := NOT_SELECTEDHANDLE.SELECTED-SUCCESSOR
IF NOT_SELECTEDHANDLE.SELECTED-SUCCESSOR = NULL-HANDLE THEN
   #DLG$WINDOW.STATUS-TEXT := 'NULL-HANDLE'
ELSE
   COMPRESS 'NEXT SELECTION: ' PTR.STRING TO #DLG$WINDOW.STATUS-TEXT
END-IF
...
```

If you want to query whether a particular item in a list box control is selected, you get the best performance by using the SELECTED attribute:

```
#DLG$WINDOW.STRING:= #LB-1-ITEMS.SELECTED(3)
```

Protecting Selection Box Controls and Input Field Controls

To prevent an end user from typing in input data in a selection box control or input field control, you have several possiblities, for example:

- setting the MODIFIABLE attribute to FALSE for the dialog element, or
- setting session parameter AD=P, or
- using a control variable (CV).

If a selection box control is protected, it is still possible to select items; only values from the item list will be displayed in its input field. If the STRING attribute is set to a value (dynamically or by initialisation) which is not in the item list, the value will not be visible to the end user.