

# Defining Mnemonic and Accelerator Keys

There are two ways of providing keyboard commands:

- A mnemonic key is determined by an underlined character in a visible dialog element, for example a menu item. The end user can select the menu item by pressing ALT+mnemonic key, for example ALT+A.
- An accelerator key is defined in the ACCELERATOR attribute. By pressing this key, the end user causes a double-click or click event for the dialog element regardless of whether the dialog element is visible or not, as long as the dialog element is enabled.

## Defining a Mnemonic Key

You define a mnemonic key in the dialog element's STRING attribute by specifying "&" before the desired character. At runtime, the character will be underlined. Example: the STRING attribute value "E&xplanation" will be displayed as "Explanation" at runtime.

If you define a mnemonic key with a text constant control or a group frame control, and the end user presses the mnemonic key at runtime, the next dialog element in the control sequence will get the focus. For example, if the next dialog element after a text constant control is an input-field control, the text constant control's mnemonic key sets the focus to the Input Field Control. Whenever you disable such an input field control at runtime, you should also disable the corresponding text constant control.

You can define mnemonic keys in the STRING attribute of the following types of dialog elements: Group Frame Control, Menu Item, Push Button Control, Radio Button Control, Text Constant Control, Toggle Button Control, Tool Bar Item.

You can still display an "&" in your runtime STRING by specifying "&&". Example: "A&&B" will be displayed as "A&B".

### Note:

In recent Windows versions (e.g. Windows 2000), mnemonic characters are, by default, not underlined until the <Alt> key is pressed. However, this new behavior can be disabled by the user, such that mnemonic characters are always underlined. For example, this can be achieved on the English version of Windows 2000 by unchecking the

"Hide Keyboard navigation indicators until I use the Alt key"

option under:

"Start/Control Panel/Display/Effects."

## Defining an Accelerator Key

You define an accelerator key by setting the ACCELERATOR attribute to a key or a key combination for the dialog element, for example to "F6" or "CTRL+1". If the end user presses the accelerator key, the double-click event occurs for the dialog element, or if no double-click event is available, the click event occurs. The accelerator key does not work if the corresponding event is suppressed, or if the dialog element is disabled.

Standard system accelerators such as "Alt+Esc", "Ctrl+Esc", "Alt+Tab" and "Ctrl+Alt+Del" can be defined as accelerators, but do not cause the dialog element's click or double-click event to be triggered. Instead, they cause the associated system functionality to be invoked. The same applies to standard MDI accelerators (such as "Ctrl+F4" and "Ctrl+F6") if used within MDI applications and to any accelerators belonging to in-place activated servers (e.g. ActiveX controls which currently have the focus).

Note that user-defined accelerator keys overwrite identical user-defined shortcut keys associated with desktop items.

If the same accelerator key is associated with more than one dialog element, the dialog element whose click or double-click event is triggered is not defined.

A dialog element which references another via its SAME-AS attribute inherits the accelerator of the referenced object. For example, if a menu item references a signal, and the signal's accelerator is "Ctrl+Alt+X", then querying the menu item's ACCELERATOR attribute will also return "Ctrl+Alt+X". However, the accelerator, if pressed, will only trigger a click event for the referenced dialog element (i.e., the signal in this example).

Accelerators of the form "Alt+X", where "X" is one of the alphabetic characters, should be avoided, because they are "reserved" for use as keyboard mnemonics.

## Displaying Accelerator Keys in Menus

In order to show accelerators for menu items, the menu text needs to first be appended with a tab (h'09') character and then appended with the text for the accelerator. This cannot be done statically in the Dialog Editor's menu editor, because there is no way to enter a tab character into the string definition. However, the accelerators may be appended dynamically using a generic piece of code which iterates round all menu items for a dialog. This is illustrated by the following external subroutine, which can conveniently be called from within a dialog's AFTER-OPEN Event.

### Example:

```

DEFINE DATA
PARAMETER
  1 #DLG$WINDOW HANDLE OF WINDOW
LOCAL
  1 #CONTROL HANDLE OF GUI
  1 #COMMAND HANDLE OF GUI
LOCAL USING NGULKEY1
END-DEFINE
*
DEFINE SUBROUTINE APPEND-ACCELERATORS
#CONTROL := #DLG$WINDOW.FIRST-CHILD
REPEAT UNTIL #CONTROL = NULL-HANDLE
  IF #CONTROL.TYPE = SUBMENU OR #CONTROL.TYPE = CONTEXTMENU
  #COMMAND := #CONTROL.FIRST-CHILD
  REPEAT UNTIL #COMMAND = NULL-HANDLE
  IF #COMMAND.ACCELERATOR <> ' '
    COMPRESS #COMMAND.STRING H'09' #COMMAND.ACCELERATOR INTO
#COMMAND.STRING LEAVING NO SPACE
  END-IF
  #COMMAND := #COMMAND.SUCCESSOR
  END-REPEAT
  END-IF
  #CONTROL := #CONTROL.SUCCESSOR
  END-REPEAT
END-SUBROUTINE
END

```

This dynamic technique has the advantage that the accelerator does not, in effect, have to be defined twice (i.e., for the ACCELERATOR and STRING attributes of the menu item).

Note that if the target language is not English, the ACCELERATOR attribute value will probably have to be translated before being appended to the menu item string.

Back to Event-Driven Programming Techniques.