

# EVENTQUALIFIER

## Applies to

Menu controls.

## Description

Sets or retrieves the event qualifier for the MENU event of a MENU item.

## Usage

*qualifier* = **Get\_Property** (*objectname*, "EVENTQUALIFIER")

*existingprop* = **Set\_Property** (*objectname*, "EVENTQUALIFIER", *qualifier*)

## Remarks

Values passed in Set\_Property and returned by both Get\_Property and Set\_Property:

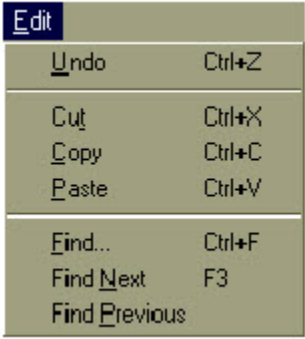
Value	Description
<i>Qualifier</i>	The Qualifier is composed of the number of parameters and the key to the SYSREPOSEVENTEXES table, separated by an asterisk. The MENU event takes only the two standard parameters, CtrlEntID and CtrlClassID. For example, the Qualifier for the MENU event for the File, Open menu item on a SYSPROG application window named "MAIN" has the following format: "2*SYSPROG*MENU*MAIN.MENU.FILE.OPEN"

## See also

[Send\\_Message\("QUALIFY\\_EVENT"\)](#)

## Example

This example shows how to create the following "Edit" menu, which will automatically enable and disable the appropriate menu items as applicable.



The *Edit* menu is named "EDIT", and the menu items are named "UNDO", "CUT", "COPY", "PASTE", "FIND", "NEXT", and "PREV". Each menu item uses an "Execute a procedure" QuickEvent; the entity names and parameters are specified in the following table:

Menu Item	Entity	Parameters
UNDO	SYSPROG*STPROCEXE**SEND_MESSAGE	'@FOCUS','UNDO'
CUT	SYSPROG*STPROCEXE**SEND_MESSAGE	'@FOCUS','CUT'
COPY	SYSPROG*STPROCEXE**SEND_MESSAGE	'@FOCUS','COPY'
PASTE	SYSPROG*STPROCEXE**SEND_MESSAGE	'@FOCUS','PASTE'
FIND	SYSPROG*STPROCEXE**SEND_MESSAGE	1,'@FOCUS'
NEXT	SYSPROG*STPROCEXE**SEND_MESSAGE	3,'@FOCUS'

PREV	SYSPROG*STPROCEXE**SEND_MESSAGE	4,'@FOCUS'
------	---------------------------------	------------

For all of the QuickEvents, make sure that the "Control" and "Property" fields in the "Return value in" section are blank.

This is the CREATE event for the window:

```
Function CREATE(CtrlEntID, CtrlClassID, CreateParam)
* the form designer does not support events for popup
* menus (those menu items which have sub-menu items), so
* use EVENTQUALIFIER to route events from the EDIT menu
* to the window's OMNIEVENT
Qualifier = 2: "***: @appid<1>: "**OMNIEVENT*": @window: "."
.Menu.Edit->EventQualifier = Qualifier
return 0
```

This is the OMNIEVENT for the window:

```
Function OMNIEVENT(CtrlEntID, CtrlClassID, Message, ...)
* if the event is generated from the "Edit" menu item,
* enable and disable the appropriate sub-menu items
declare function EditFind
$insert EditFind_Equates
$insert Logical

begin case
case CtrlEntID [-1,"B."] = "EDIT"
* get the current focus
Focus = @@window->Focus
* check if the control type is paste-able
locate @Focus->Type in "EDITFIELD,EDITTABLE,EDITBOX,COMBOBOX"
using "," setting Pos then
CanPaste = TRUE$
end else
CanPaste = FALSE$
end
* enable/disable menu items as appropriate
.Menu.Edit.Undo->Enabled = @Focus->CanUndo
.Menu.Edit.Cut->Enabled = |
(abs(@Focus->Selection<2>) > 0)
.Menu.Edit.Copy->Enabled = .Menu.Edit.Cut->Enabled
.Menu.Edit.Paste->Enabled = |
(CanPaste and len(ClipBoard-Text))
.Menu.Edit.Find->Enabled = |
EditFind(EFCMD_ISFINDABLE$,Focus)
.Menu.Edit.Next->Enabled = |
EditFind(EFCMD_ISFINDNEXTABLE$,Focus)
.Menu.Edit.Prev->Enabled = .Menu.Edit.Next->Enabled
end case
return 0
```