
Subject: Re: New TIMER keyword in widget_control

Posted by [oet](#) on Wed, 16 Jun 1993 06:19:30 GMT

[View Forum Message](#) <> [Reply to Message](#)

In article 1vc5gaINN1ra@maz4.sma.ch, oet@sma.ch (Thomas Oettli) writes:

> IDL 3.1:

>

> Has someone already used the new TIMER keyword in widget_control?

>

> The old xmanager background task syntax BACKGROUND='<background proc>'

> is still working but produces the message

>

> "XMANAGER: The BACKGROUND keyword to theXMANAGER procedure is obsolete. It is

> superceeded by the TIMER keyword to the WIDGET_CONTROL procedure."

>

> I couldn't find enough documentation in the IDL Distribution 3.1 for how to

> change code to use this new enhanced feature for background operations.

> The background task example in \$IDL_DIR/lib/xdemo/examples, wback.pro

> uses still the old xmanager syntax.

>

sorry for following up myself, I've just got an answer from the RSI support.

Below the changed code example for the wback procedure in

\$IDL_DIR/lib/xdemo/examples. It seems to be much easier now to change the delay time

using a single widget_control statement instead of if-expired-time-checkings

in the background procedure.

-Thomas

```
;* ----- WTIMER.PRO ----- CUT HERE -----
```

```
; This is the code for a widget that performs a task at given time
```

```
; intervals.
```

```
; Here, a text window appears and the current system time is displayed
```

```
; repeatedly. Select the "Done" button to exit.
```

```
PRO wtimer_task
```

```
; This routine does the timer task when called from the event handler
```

```
; of the main routine.
```

```
; The COMMON block is used because the timer task needs
```

```
; the widget id of the text widget:
```

```
COMMON wtimerblock, text1, base
```

; This is the task that the widget performs:

```
temp_string = SYSTIME(0)
WIDGET_CONTROL, text1, SET_VALUE=temp_string, /APPEND

END
```

```
PRO wtimer_event, event
```

; This is the event handler which will also handle a timer widget.

; The COMMON block is used because the event handler needs the
; widget id of the base which has registered a timer:

```
COMMON wtimerblock, text1, base
```

; If a widget has been selected, put its User Value into 'eventval':

```
WIDGET_CONTROL, event.id, GET_UVALUE = eventval
```

```
IF (event.id EQ base) then begin
```

```
    ; call the timer task
```

```
    wtimer_task
```

```
    ; request another timer event, so it continues to generate timer events
```

```
    WIDGET_CONTROL, base, TIMER=0.0
```

```
endif else begin
```

```
    ; Perform actions based on the user value of the event:
```

```
    CASE eventval OF
```

```
        'DONE' : WIDGET_CONTROL, event.top, /DESTROY
```

```
        'ERASE': WIDGET_CONTROL, text1, SET_VALUE = "
```

```
    ENDCASE
```

```
endelse
```

```
END
```

```
PRO wtimer, GROUP=GROUP
```

; This is the procedure that creates a widget which has a timer event.

; The COMMON block is used because the event handler needs
; the widget id of the text widget:

COMMON wtimerblock, text1, base

; A top-level base widget with the title "Timer Widget Example"
; is created:

base = WIDGET_BASE(TITLE = 'Timer Widget Example', \$
/COLUMN)

WIDGET_CONTROL, base, TIMER=5.0 ;generate a timer event after 5 seconds.
;use this base, since it does not generate
;any other events.
;a label widget could also be used.

; Make the 'DONE' button:

button1 = WIDGET_BUTTON(base, \$
UVALUE = 'DONE', \$
VALUE = 'DONE')

; Make the text widget:

text1 = WIDGET_TEXT(base, \$; create a display only text widget
XSIZE=30, \$
YSIZE=30, \$
/SCROLL)

; Make a button which will clear the text file.

button2 = WIDGET_BUTTON(base, \$
UVALUE = 'ERASE', \$
VALUE = 'ERASE')

; Realize the widgets:

WIDGET_CONTROL, base, /REALIZE

; Hand off control of the widget to the XMANAGER
XMANAGER, "wtimer", base, GROUP_LEADER=GROUP

END

; *----- END WTIMER.PRO CUT HERE -----
