
Subject: Re: Running processes in parallel

Posted by [Michael Galloy](#) on Wed, 12 Jul 2006 20:25:00 GMT

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

WorkerAnt wrote:

> Currently, there is a for loop in my program that takes anywhere from a
> couple of seconds to a couple of minutes. (The procedure runs a step by
> step animation using the wait function). It's called from an event
> handler whenever a button in the widget is pushed. I want to be able to
> stop anytime in the middle of the animation when another button is
> pressed. (As opposed to the ctrl break method).
> However, the event handler is unable to process the stop button getting
> pushed until the for loop procedure is complete, thus defeating the
> purpose. Is there any way to run two procedures in parallel? Or a time
> independent way to have one take priority over another?
> Barring these two possibilities working, is there any other way of
> having the animation stop?
>

Yes there is a way to do this: use timer events. You'll need someplace to store data (like a "state" structure, member variables of an object, etc.) The basic steps would be, in your current event handler:

1. If the event was a "stop" button, then set a "stop" flag and exit.
2. If the "stop" flag is set, exit.
2. Do one step and record that you did the step (so you know which step to do the next time).
3. Set a timer (using "WIDGET_CONTROL, id, TIMER=t" instead of using "WAIT, t").

You could probably do this using the IDL_IDLBridge now, in a way that is more in line with your original strategy. I don't have a lot of experience using it, but it seems like there is a fair amount of overhead in using it.

Mike

--

www.michaelgalloy.com

Subject: Re: Running processes in parallel

Posted by [David Fanning](#) on Wed, 12 Jul 2006 20:44:22 GMT

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

WorkerAnt writes:

> Currently, there is a for loop in my program that takes anywhere from a
> couple of seconds to a couple of minutes. (The procedure runs a step by

- > step animation using the wait function). It's called from an event
- > handler whenever a button in the widget is pushed. I want to be able to
- > stop anytime in the middle of the animation when another button is
- > pressed. (As opposed to the ctrl break method).
- > However, the event handler is unable to process the stop button getting
- > pushed until the for loop procedure is complete, thus defeating the
- > purpose. Is there any way to run two procedures in parallel? Or a time
- > independent way to have one take priority over another?
- > Barring these two possibilities working, is there any other way of
- > having the animation stop?

You can see an example of using timer events to run a widget animation in this example program:

<http://www.dfanning.com/programs/xmovie.pro>

Cheers,

David

--

David Fanning, Ph.D.

Fanning Software Consulting, Inc.

Coyote's Guide to IDL Programming: <http://www.dfanning.com/>

Subject: Re: Running processes in parallel

Posted by [Roberto Monaco](#) on Wed, 12 Jul 2006 23:12:46 GMT

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

I had a similar problem (to stop a time consuming process by pressing a button) and wanted to experiment with IDL_IDLBridge, so created a program that calls David's progressbar and sets a shared memory flag when the "cancel" button is pressed. It is called through a IDL_IDLbridge (as a separate process) before entering the loop, for example:

```
oBridge = OBJ_NEW("IDL_IDLBridge")    ; create the child process

; create shared memory (2 elements array: a flag and the value to pass
; to progressbar to show progress)
SHMMAP, 'progressbar_shm', 2, /INTEGER
shm_var = shmvar('progressbar_shm')
shm_var[0] = 0

; launch a progress bar
oBridge->EXECUTE, "progressbar_bridge", /NOWAIT
```

```

; the loop ends if shm_var[0] is set (the cancel button was pressed)
i = 0
WHILE i LT 10 AND NOT shm_var[0] DO BEGIN
    shm_var[1] = i*10.0
    i = i + 1
    ; here comes the time consuming thing inside the loop (simulated by
wait)
    WAIT, 10
ENDWHILE

;; destroy the object and unmap shared memory
OBJ_DESTROY, oBridge
SHMUNMAP, 'progressbar_shm'

PRO progressbar_bridge, TEXT=text

; create a 2-element fix array shared memory block
; progressbar_shm[0] = flag (=1 if <CANCEL> was pressed, =0 otherwise)
; progressbar_shm[1] = progress (the fix number to update the
progressbar)
SHMMAP, 'progressbar_shm', 2, /INTEGER
shm_var = SHMVAR('progressbar_shm') ;
shm_var[1] = 0 ; starts with progress = 0%

pbar = OBJ_NEW('progressbar', TEXT=text)
pbar->Start

WHILE NOT shm_var[0] DO BEGIN
    pbar->Update, shm_var[1]
    IF pbar->CheckCancel() THEN $
        shm_var[0] = 1 ; set flag=1
    WAIT, 0.1 ; refresh every 10th of second
ENDWHILE
pbar->Destroy
SHMUNMAP, 'progressbar_shm'

END

```

END

I had done some adaptation to progressbar time ago, so I am not sure this works with the original... anyhow I hope it helps or gives some ideas.

Roberto

"Michael Galloy" <mgalloy@gmail.com> wrote in message
news:7Ladnc67C4kgxyjZnZ2dnUVZ_tWdnZ2d@comcast.com...
> WorkerAnt wrote:

>> Currently, there is a for loop in my program that takes anywhere from a
>> couple of seconds to a couple of minutes. (The procedure runs a step by
>> step animation using the wait function). It's called from an event
>> handler whenever a button in the widget is pushed. I want to be able to
>> stop anytime in the middle of the animation when another button is
>> pressed. (As opposed to the ctrl break method).
>> However, the event handler is unable to process the stop button getting
>> pushed until the for loop procedure is complete, thus defeating the
>> purpose. Is there any way to run two procedures in parallel? Or a time
>> independent way to have one take priority over another?
>> Barring these two possibilities working, is there any other way of
>> having the animation stop?
>>
>
> Yes there is a way to do this: use timer events. You'll need someplace to
> store data (like a "state" structure, member variables of an object, etc.)
> The basic steps would be, in your current event handler:
>
> 1. If the event was a "stop" button, then set a "stop" flag and exit.
> 2. If the "stop" flag is set, exit.
> 2. Do one step and record that you did the step (so you know which step
> to do the next time).
> 3. Set a timer (using "WIDGET_CONTROL, id, TIMER=t" instead of using
> "WAIT, t").
>
> You could probably do this using the IDL_IDLBridge now, in a way that is
> more in line with your original strategy. I don't have a lot of experience
> using it, but it seems like there is a fair amount of overhead in using
> it.
>
> Mike
> --
> www.michaelgalloy.com
