Subject: Evenly timed events in a widget program
Posted by dg86 on Sun, 15 Sep 2013 16:33:09 GMT
View Forum Message <> Reply to Message

Dear Folks,

I'm working on a widget program that resembles a TV screen with icons overlaid on the video. In my present implementation, a widget timer event triggers updates of the TV screen, while users interacting with the icons generate other widget events. My problem is that the timer events can get stacked up behind a great many user-generated events, causing the TV screen to update erratically.

Is there a way to convince xmanager to act on timer events as soon as they arrive, rather than working its way systematically but indiscriminately through the stack of events?

Alternatively, is there a way to run the TV widget as some sort of background process that runs on its own fixed schedule independent of the user interaction in the foreground?

If the best answer is to ditch xmanager and to call widget_event directly, can anyone point me to an effectively written example that I might use as a model for my own code?

Many thanks,

David

Subject: Re: Evenly timed events in a widget program
Posted by David Fanning on Sun, 15 Sep 2013 16:45:24 GMT
View Forum Message <> Reply to Message

David Grier writes:

> I'm working on a widget program that resembles a TV screen with icons overlaid on the
> video. In my present implementation, a widget timer event triggers updates of the TV screen,
> while users interacting with the icons generate other widget events. My problem is that the
timer
> events can get stacked up behind a great many user-generated events, causing the TV screen
to
> update erratically.
>
> Is there a way to convince xmanager to act on timer events as soon as they
> arrive, rather than working its way systematically but indiscriminately through
> the stack of events?
>
> Alternatively, is there a way to run the TV widget as some sort of background process
> that runs on its own fixed schedule independent of the user interaction in the foreground?

>
> If the best answer is to ditch xmanager and to call widget_event directly,
> can anyone point me to an effectively written example that I might use as a
> model for my own code?

I would say the answer is probably no on all accounts. :-(

You can certainly clear events (CLEAR_EVENTS keyword to Widget_Control), but it doesn't sound as though you really want to. The controls are probably there for a reason. I don't know any way to force an event to the head of the queue, like the VIP pass at Disney World does.

Running the TV widget as a background process is an interesting idea. What if you wrote it as a widget object and then controlled it from the IDL bridge? Doubt it will work, but I would be interested in the result if you try it. :-)

Ditching the XManager is almost always a bad idea, fraught with unexpected consequences, usually.

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thue. ("Perhaps thou speakest truth.")

Subject: Re: Evenly timed events in a widget program
Posted by dg86 on Mon, 16 Sep 2013 01:43:41 GMT
View Forum Message <> Reply to Message

On Sunday, September 15, 2013 12:45:24 PM UTC-4, David Fanning wrote:
> David Grier writes:
>
>
>
>> I'm working on a widget program that resembles a TV screen with icons overlaid on the
>
>> video. In my present implementation, a widget timer event triggers updates of the TV screen,
>
>> while users interacting with the icons generate other widget events. My problem is that the timer

>
>> events can get stacked up behind a great many user-generated events, causing the TV screen to
>
>> update erratically.
>
>>
>
>> Is there a way to convince xmanager to act on timer events as soon as they
>
>> arrive, rather than working its way systematically but indiscriminately through
>
>> the stack of events?
>
>>
>
>> Alternatively, is there a way to run the TV widget as some sort of background process
>
>> that runs on its own fixed schedule independent of the user interaction in the foreground?
>
>>
>
>> If the best answer is to ditch xmanager and to call widget_event directly,
>
>> can anyone point me to an effectively written example that I might use as a
>
>> model for my own code?
>
>
>
> I would say the answer is probably no on all accounts. :-(
>
>
>
> You can certainly clear events (CLEAR_EVENTS keyword to Widget_Control),
>
> but it doesn't sound as though you really want to. The controls are
>
> probably there for a reason. I don't know any way to force an event to
>
> the head of the queue, like the VIP pass at Disney World does.
>
>
>
> Running the TV widget as a background process is an interesting idea.
>
> What if you wrote it as a widget object and then controlled it from the
>

> IDL bridge? Doubt it will work, but I would be interested in the result
>
> if you try it. :-)
>
>
>
> Ditching the XManager is almost always a bad idea, fraught with
>
> unexpected consequences, usually.
>
>
>
> Cheers,
>
>
>
> David
>
>
>
>
>
>
>
>
> --
>
> David Fanning, Ph.D.
>
> Fanning Software Consulting, Inc.
>
> Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
>
> Sepore ma de ni thue. ("Perhaps thou speakest truth.")

Thanks very much, David, for advising against rolling my own event manager.

Lacking better ideas, I tried a hideous hack that works reasonably well.
My event handler now calls systime(1) for every event, including timer events, and updates the video screen whenever the intended time interval has passed. Screen updates no longer are tied rigorously to widget timer events, but rather are tied to the system clock. Widget timer events ensure that the event loop looks at the clock on a regular basis.

In the same spirit, I added the screen update routine as a callback for time-consuming operations triggered by user interactions.

Having the screen run as a background process would have been much more elegant, but I could not figure out how to get an IDL_IDLBridge to update an "outside" widget.

Now that IDL is acquiring video capabilities, it'd be cool to have a video object that can be embedded in an IDLgrScene and runs smoothly in the background.

All the best,

David
