Subject: testing IDL_IDLBridge status
Posted by markb77 on Thu, 05 Nov 2015 10:49:06 GMT
View Forum Message <> Reply to Message

hi,

I have a question about how to query the status of an IDL bridge, in order to know when to destroy the bridge process.

My IDL application is written as an object, and it is started by creating an "application" object. After creation, it might carry out CPU-intensive tasks like processing a stack of 10^6 images, for example.

I want to be able to run multiple instances of my application which can operate in parallel, and the only way to do this is to create each instance on a separate IDL process, i.e. a separate IDL Bridge object.

I want to automatically clean up these bridge objects when the application closes. When the user closes one instance of the application, they are essentially just destroying the application object.

One way to test when I should destroy the bridge object is to periodically query the bridge to see if the object reference to the application is still valid. However, if the application is busy (i.e. processing data) my query of the bridge (by executing 'test=obj_valid(application_object)') will hang.

The "status" property of the bridge doesn't seem to help here. Even if the application object is busy and the bridge won't respond to a GetVar command, the "status" property of the bridge still reports 0 (IDLE).

How can I test when to destroy the bridge? What I want is a main-level process that is periodically checking the state of the application objects on the various bridges that have been created. If the application object has been closed, then destroy the bridge. Right now I can find no way of querying the bridges without getting into a situation where my main-level process will get stuck if the application is busy.

any ideas?
thanks
Mark

Subject: Re: testing IDL_IDLBridge status
Posted by markb77 on Thu, 05 Nov 2015 13:32:20 GMT
View Forum Message <> Reply to Message

After further investigation, it seems that there is really no way to do this. When the application object is busy, even though the status of the bridge object remains 0 or 2 (Idle or Command completed), the bridge object will not respond to the Execute method. I need to use Execute in order to query the status of the application.

For example: (pseudocode)

```
my_bridge = obj_new('IDL_IDLBridge')

cmd = 'my_app = obj_new("my_application")'
my_bridge -> Execute, cmd

test_quit = 0

while test_quit eq 0 do begin

   cmd = 'test_app_closed = ~obj_valid(my_app)'
   my_bridge -> Execute, cmd

   test_app_closed = my_bridge -> GetVar('test_app_closed')

   if test_app_closed eq 1 then begin

      obj_destroy, my_bridge
      test_quite = 1

   endif

endwhile
```

This doesn't work, because when the application is busy, the program gets stuck at the Execute statement. If this program is running in the main IDL process and gets stuck, then I can't do anything else in IDL while I'm waiting for the application to finish its task.

If I use the NOWAIT keyword to Execute, and specify a callback procedure etc., it makes no difference. The program still gets stuck at the Execute statement.

The "STATUS" property of the bridge is constantly reporting either 0 or 2 (IDLE or COMMAND COMPLETE) throughout this process. Even though the application is busy and is effectively blocking the bridge from doing any other processing, the status reports 0 or 2. Effectively, it is impossible for my program to query the bridge without getting hung up!

Therefore, it seems there is no way for my to automatically kill the IDL bridge processes when the application objects are closed. Unless, there is a way for the application object to kill the IDL bridge process on its own, from within IDL?

ideas?
thanks

Mark

Subject: Re: testing IDL_IDLBridge status
Posted by Jim Pendleton on Thu, 05 Nov 2015 13:54:46 GMT
View Forum Message <> Reply to Message

On Thursday, November 5, 2015 at 6:32:24 AM UTC-7, superchromix wrote:
> After further investigation, it seems that there is really no way to do this. When the application object is busy, even though the status of the bridge object remains 0 or 2 (Idle or Command completed), the bridge object will not respond to the Execute method. I need to use Execute in order to query the status of the application.
>
> For example: (pseudocode)
>
> ---
>
> my_bridge = obj_new('IDL_IDLBridge')
>
> cmd = 'my_app = obj_new(''my_application'')'
> my_bridge -> Execute, cmd
>
> test_quit = 0
>
> while test_quit eq 0 do begin
>
> cmd = 'test_app_closed = ~obj_valid(my_app)'
> my_bridge -> Execute, cmd
>
> test_app_closed = my_bridge -> GetVar('test_app_closed')
>
> if test_app_closed eq 1 then begin
>
> obj_destroy, my_bridge
> test_quite = 1
>
> endif
>
> endwhile
>
> ---
>
> This doesn't work, because when the application is busy, the program gets stuck at the Execute statement. If this program is running in the main IDL process and gets stuck, then I can't do anything else in IDL while I'm waiting for the application to finish its task.
>
> If I use the NOWAIT keyword to Execute, and specify a callback procedure etc., it makes no difference. The program still gets stuck at the Execute statement.

>
> The "STATUS" property of the bridge is constantly reporting either 0 or 2 (IDLE or COMMAND COMPLETE) throughout this process. Even though the application is busy and is effectively blocking the bridge from doing any other processing, the status reports 0 or 2. Effectively, it is impossible for my program to query the bridge without getting hung up!
>
> Therefore, it seems there is no way for my to automatically kill the IDL bridge processes when the application objects are closed. Unless, there is a way for the application object to kill the IDL bridge process on its own, from within IDL?
>
> ideas?
> thanks
> Mark

Which platform and version of IDL are you using?

Subject: Re: testing IDL_IDLBridge status
Posted by markb77 on Thu, 05 Nov 2015 14:21:55 GMT
View Forum Message <> Reply to Message

IDL> print, !version
{ x86_64 Win32 Windows Microsoft Windows 8.5 Jul 7 2015 64 64}

Subject: Re: testing IDL_IDLBridge status
Posted by markb77 on Thu, 05 Nov 2015 15:08:27 GMT
View Forum Message <> Reply to Message

One possible workaround to this would be for the application object itself to kill the bridge object on which it is running.

However, calling "EXIT, STATUS=0" from within my application object (in the Cleanup procedure) seems to have no effect?

Subject: Re: testing IDL_IDLBridge status
Posted by markb77 on Thu, 05 Nov 2015 15:16:14 GMT
View Forum Message <> Reply to Message

Even better: if I use the EXECUTE function with EXIT, it still doesn't work.

```
cmdstr = 'EXIT, STATUS=0'
result = EXECUTE(cmdstr)
```

the EXECUTE statement has no effect on IDL when executed from within the cleanup procedure of my application.

???

Subject: Re: testing IDL_IDLBridge status
Posted by Dominic[2] on Thu, 05 Nov 2015 17:00:08 GMT
View Forum Message <> Reply to Message

On Thursday, November 5, 2015 at 10:16:17 AM UTC-5, superchromix wrote:
> Even better: if I use the EXECUTE function with EXIT, it still doesn't work.
>
> cmdstr = 'EXIT, STATUS=0'
> result = EXECUTE(cmdstr)
>
> the EXECUTE statement has no effect on IDL when executed from within the cleanup procedure of my application.
>
> ???

Hi Mark,

One solution is to check the status in your callback function. The callback function will also have a reference to your bridge object as its third argument :

PRO BridgeCallbackName, Status, Error, Objref [, Userdata]

Depending upon the status value, you can destroy the object from within the callback. For example, for a successful completion (status=2), enter the following line before returning from your callback function:

if status eq 2 then obj_destroy,Objref

Dominic

Subject: Re: testing IDL_IDLBridge status
Posted by Jim Pendleton on Fri, 06 Nov 2015 02:02:28 GMT
View Forum Message <> Reply to Message

On Thursday, November 5, 2015 at 6:32:24 AM UTC-7, superchromix wrote:
> After further investigation, it seems that there is really no way to do this. When the application object is busy, even though the status of the bridge object remains 0 or 2 (Idle or Command completed), the bridge object will not respond to the Execute method. I need to use Execute in order to query the status of the application.
>
> For example: (pseudocode)
>

```
>  ------------------------------
>
>  my_bridge = obj_new('IDL_IDLBridge')
>
>  cmd = 'my_app = obj_new(''my_application'')'
>  my_bridge -> Execute, cmd
>
>  test_quit = 0
>
>  while test_quit eq 0 do begin
>
>     cmd = 'test_app_closed = ~obj_valid(my_app)'
>     my_bridge -> Execute, cmd
>
>     test_app_closed = my_bridge -> GetVar('test_app_closed')
>
>     if test_app_closed eq 1 then begin
>
>        obj_destroy, my_bridge
>        test_quite = 1
>
>     endif
>
>  endwhile
>
>  ------------------------------
>
```
> This doesn't work, because when the application is busy, the program gets stuck at the Execute statement. If this program is running in the main IDL process and gets stuck, then I can't do anything else in IDL while I'm waiting for the application to finish its task.
>
> If I use the NOWAIT keyword to Execute, and specify a callback procedure etc., it makes no difference. The program still gets stuck at the Execute statement.
>
> The "STATUS" property of the bridge is constantly reporting either 0 or 2 (IDLE or COMMAND COMPLETE) throughout this process. Even though the application is busy and is effectively blocking the bridge from doing any other processing, the status reports 0 or 2. Effectively, it is impossible for my program to query the bridge without getting hung up!
>
> Therefore, it seems there is no way for my to automatically kill the IDL bridge processes when the application objects are closed. Unless, there is a way for the application object to kill the IDL bridge process on its own, from within IDL?
>
> ideas?
> thanks
> Mark
```

You might consider using a non-blocking bridge EXECUTE call, locking a semaphore
```

(SEM_LOCK) on the bridge side before the execution of long-running jobs, and checking the state of the semaphore through a timer() in the main process.

It's best to stop and close bridge processes from the main IDL process that created them rather than via EXIT on the bridge side. The originating IDL process is a little touchy about the object lifecycle and references may be left in limbo.

Subject: Re: testing IDL_IDLBridge status
Posted by markb77 on Fri, 06 Nov 2015 08:50:22 GMT
View Forum Message <> Reply to Message

On Thursday, November 5, 2015 at 6:00:12 PM UTC+1, dmzarro wrote:
> On Thursday, November 5, 2015 at 10:16:17 AM UTC-5, superchromix wrote:
>> Even better: if I use the EXECUTE function with EXIT, it still doesn't work.
>>
>> cmdstr = 'EXIT, STATUS=0'
>> result = EXECUTE(cmdstr)
>>
>> the EXECUTE statement has no effect on IDL when executed from within the cleanup procedure of my application.
>>
>> ???
>
> Hi Mark,
>
> One solution is to check the status in your callback function. The callback function will also have a reference to your bridge object as its third argument :
>
> PRO BridgeCallbackName, Status, Error, Objref [, Userdata]
>
> Depending upon the status value, you can destroy the object from within the callback. For example, for a successful completion (status=2), enter the following line before returning from your callback function:
>
> if status eq 2 then obj_destroy,Objref
>
> Dominic

hi Dominic,

Thanks, but this solution doesn't work. When the application is started on the bridge, using a command similar to:

my_app = obj_new('my_application')

this command executes normally and the bridge status is set to 2 (command completed). However, if you kill the bridge now, you would kill the whole application. What you need to do is

wait until the application is closed by the user (it is a GUI application), and only then can you destroy the bridge.

The question is, how do you check if the application is closed? Any call to Execute or Execute, /NOWAIT will get stuck if the application is busy, even though the bridge status indicates 0 or 2.

thoughts?

Subject: Re: testing IDL_IDLBridge status
Posted by markb77 on Fri, 06 Nov 2015 09:19:25 GMT
View Forum Message <> Reply to Message

On Friday, November 6, 2015 at 3:02:35 AM UTC+1, Jim P wrote:
> On Thursday, November 5, 2015 at 6:32:24 AM UTC-7, superchromix wrote:
>> After further investigation, it seems that there is really no way to do this. When the application object is busy, even though the status of the bridge object remains 0 or 2 (Idle or Command completed), the bridge object will not respond to the Execute method. I need to use Execute in order to query the status of the application.
>>
>> For example: (pseudocode)
>>
>> ---
>>
>> my_bridge = obj_new('IDL_IDLBridge')
>>
>> cmd = 'my_app = obj_new(''my_application'')'
>> my_bridge -> Execute, cmd
>>
>> test_quit = 0
>>
>> while test_quit eq 0 do begin
>>
>> cmd = 'test_app_closed = ~obj_valid(my_app)'
>> my_bridge -> Execute, cmd
>>
>> test_app_closed = my_bridge -> GetVar('test_app_closed')
>>
>> if test_app_closed eq 1 then begin
>>
>> obj_destroy, my_bridge
>> test_quite = 1
>>
>> endif
>>
>> endwhile
>>
>> ---

>>
>> This doesn't work, because when the application is busy, the program gets stuck at the Execute statement. If this program is running in the main IDL process and gets stuck, then I can't do anything else in IDL while I'm waiting for the application to finish its task.
>>
>> If I use the NOWAIT keyword to Execute, and specify a callback procedure etc., it makes no difference. The program still gets stuck at the Execute statement.
>>
>> The "STATUS" property of the bridge is constantly reporting either 0 or 2 (IDLE or COMMAND COMPLETE) throughout this process. Even though the application is busy and is effectively blocking the bridge from doing any other processing, the status reports 0 or 2. Effectively, it is impossible for my program to query the bridge without getting hung up!
>>
>> Therefore, it seems there is no way for my to automatically kill the IDL bridge processes when the application objects are closed. Unless, there is a way for the application object to kill the IDL bridge process on its own, from within IDL?
>>
>> ideas?
>> thanks
>> Mark
>
> You might consider using a non-blocking bridge EXECUTE call, locking a semaphore (SEM_LOCK) on the bridge side before the execution of long-running jobs, and checking the state of the semaphore through a timer() in the main process.
>
> It's best to stop and close bridge processes from the main IDL process that created them rather than via EXIT on the bridge side. The originating IDL process is a little touchy about the object lifecycle and references may be left in limbo.

hi Jim,

I think the workaround using semaphores might work - I can lock a semaphore on the bridge side when the application starts, and then the application will release the semaphore before it closes. Then in the main process I will kill the bridge when the semaphore becomes free.

It's a bit of a backwards way of doing things, though. It seems to me that the status property of the bridge object should more accurately reflect whether the bridge is busy or not.

thanks for the suggestion.
Mark

Subject: Re: testing IDL_IDLBridge status
Posted by markb77 on Fri, 06 Nov 2015 14:10:12 GMT
View Forum Message <> Reply to Message

Just as an update, this solution worked. Semaphores are a useful means of communication

between the main IDL process and the bridge objects, without necessarily requiring that the bridges process is currently available for communication (i.e. IDLE).