Subject: Alternative to EXECUTE?
Posted by anne.martel on Tue, 02 Dec 2003 14:40:31 GMT
View Forum Message <> Reply to Message

I've just realised that the IDL Virtual Machine won't work with EXECUTE. Does anyone have an alternative for the sort of use described below?

I'm using EXECUTE to initialise an object with a set of fields defined by the structure info. The number of fields defined in info will vary depending on the type of image I'm reading in so I only want to set a subset of the fields. I could explicitity test for each tag name but that makes it harder to add new fields later and I would end up with a very long CASE statement.

```
pro QmcImage::set_info,info
   tmp_class = {QmcImage}
   object_fields = Tag_Names(tmp_class)
   info_fields = Tag_Names(info)
   num=n_elements(info_fields)
   for i = 0,num-1  do begin
      index = where(strcmp( info_fields[i], object_fields,/F),count)
     if(count eq 1) then begin
        str='self.' + info_fields[i] + '= info.' +  info_fields[i]
         ok = execute(str)
      endif
   endfor
end
```

Thanks,

Anne Martel
Imaging Research,
Sunnybrook and Women's CHSC

Subject: Re: alternative to execute
Posted by Robert Barnett on Tue, 31 Jan 2006 22:02:45 GMT
View Forum Message <> Reply to Message

Writing out 300 functions would seen like a bit of a drain, but it might be your only choice if you cannot see any pattern across the variation of these functions. Presuming there are some patterns to simplify the problem then I'd probably use object oriented design. I've been in similar situations before and I've found OO often helps you implement the patterns.

The way I approached my similar problem was to:

Step 1: Create a base class which has common methods and fields across all of your functions.

```
pro arithmetic__define, struct
struct = {arithmetic, field1: ptr_new(), field2: ptr_new(), field3:
ptr_new()}
end
```

Step 2: Create class definitions for every function. (Use a perl script or something to generate the code)

```
pro artimetic1__define, struct
struct = {arithmetic1, INHERITS arithmetic}
end

pro artimetic2__define, struct
struct = {arithmetic2, INHERITS arithmetic}
end

pro artimetic3__define, struct
struct = {arithmetic3, INHERITS arithmetic}
end
```

Step 3: Implement your functions across all classes

```
pro artimetic1::solve, b123, b100, b050, b035, RESULT=result
result = b123 / b100 - *self.field1 * b050 / b035
end

pro artimetic2::solve, b123, b050, b035, RESULT=result
result = b123 - *self.field1 * b050 / b035
end

pro artimetic3::solve, b123, b050, b035, RESULT=result
result = b123 - b050 / *self.field1 * b035
end
```

The advantage of this style of programming is that you can use inheritance to tweak functions. Take your function, "arithmetic2". You could create two tweaked version "arithmetic2a" and "arithmetic2b".

For example:

```
pro artimetic2::solve, b123, b050, b035, RESULT=result
result = b123 - *self.field1 * self -> apply(b050, b035 )
end
```

```
pro artimetic2a__define, struct
struct = {arithmetic2a, INHERITS arithmetic2}
end

pro artimetic2b__define, struct
struct = {arithmetic2b, INHERITS arithmetic2}
end

function artimetic2a::apply, a, b
return, a*b
end

function artimetic2b::apply, a, b
return, a/b
end
```

I've really only skimmed the surface of OO design here. For brevity I haven't shown the init, cleanup or Get/Set property methods. But, hopefully it is food for thought.

P.S. You could half the code written here if CREATE_STRUCT accepted the INHERITS keyword.

Robbie

Subject: Re: alternative to execute
Posted by Ken Mankoff on Wed, 01 Feb 2006 02:22:26 GMT
View Forum Message <> Reply to Message

On Tue, 31 Jan 2006, greg michael wrote:
> I'm trying to think up a good way to implement a user-defined
> function which would run in the IDL VM. The function might involve
> any of about 300 similar 2-d arrays, each of which is quite
> expensive to generate. So I thought to obtain the function as a
> string, scan it to see which arrays are needed, generate them, and
> then apply the function with 'execute'. But 'execute' is excluded
> from the VM.
>
> A typical function might look like:
>
> result = b123 / b100 - .9 * b050 / b035
>
> Without execute, I can only think to attempt to write some kind of
> arithmetic interpreter. Would anyone have a suggestion for a way
> to get IDL to do this more directly?

>

Dump the data to disk, generate a small PRO that reads in the data, executes your user function (using execute), and then dumps the result back to disk, spawn a new 2nd IDL session, not VM, that runs the generated PRO. As long as it takes less than 7 minutes to run everything should work. Read back in the results, and continue.

-k.

Subject: Re: alternative to execute
Posted by marc schellens[1] on Wed, 01 Feb 2006 08:09:00 GMT
View Forum Message <> Reply to Message

IDL in trial mode does not allow any reading to or writing from files.
Marc

Subject: Re: alternative to execute
Posted by greg michael on Wed, 01 Feb 2006 11:01:37 GMT
View Forum Message <> Reply to Message

Thanks very much, Robbie, for your detailed reply. Unfortunately, it's not 300 functions that I have, but 300 parameters. I couldn't enumerate the set of possible functions. So I don't think I could apply your method (if I've understood it correctly).

I got a suggestion by email (thanks, Matt) to use call_function.

What I don't have a solution for, is how to handle operator precedence and simple arithmetic operators -

e.g. result = (b001 + b002) / b003 + sqrt(b004) + b005^2 (unrealistic, but demonstrative example)

'execute' would have managed all that for me. If I do it using call_function, it looks to me like I'll have to write code to pull out the brackets, apply each type of operator and function in the right order. call_function would handle only the 'sqrt' (when I got that far) - or am I wrong?

Maybe it's not such a difficult task to write this code, but a pity since IDL already knows how to do it!

greg

Subject: Re: alternative to execute
Posted by greg michael on Wed, 01 Feb 2006 11:04:12 GMT
View Forum Message <> Reply to Message

Thanks for this exotic solution, but I'm afraid, as Marc says, the file access would be a problem.

greg