

---

Subject: Re: IDLVM and execute

Posted by [Michael Galloy](#) on Sun, 22 Jan 2012 19:59:18 GMT

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

---

On 1/22/12 8:30 am, Russell wrote:

> On Jan 20, 3:36 pm, TonyLanz<tolan...@gmail.com> wrote:

>> Hi all,

>> I realize this has been asked on the group before. I'm hoping someone  
>> might have a "simple" workaround in IDLVM to what would be easily  
>> accomplished using the execute function in IDL proper. I have a widget  
>> program that contains a compound widget field that allows the user to  
>> define an equation for manipulating a set of four variables. For the  
>> sake of argument let's say var1, var2, var3, and var4. In this  
>> compound widget field they can define what math they want to do,  
>> they're restricted to addition, subtraction, multiplication and  
>> division.

>>

>> so they could for example enter

>>

>> var1/var2

>> or

>> var1-var2\*0.2

>> or

>> var1+var2+var3

>>

>> etc., you get the idea. Now in IDL I can just get the value from the  
>> widget and simply do something like

>>

>> s='result='+cwidget\_value(0)

>> status=execute(s)

>> print,result

>>

>> of course in IDLVM I'm not allowed to use EXECUTE()

>>

>> Anyone have any suggestions for implementing this simply? I know in  
>> the past there was discussion of writing code to break down the string  
>> and pass the pieces to the right operators or functions (using  
>> call\_function() ) to assemble the result.

>>

>> Tony

>

> Yes, like all things, there are several ways to skin this cat. The  
> simplest way is to write the string to a file, compile that file, then  
> call it as a function (and presumably delete it after you're  
> finished). So if you have the command as a string, such as "result =  
> var1 + var2\*0.2" then just write it to a file with the appropriate  
> declarations and closings. So, you'd execute:

>

```

> ;save the data:
> vars=[var1,var2]
>
> ;write the file:
> openw,lun,'file.pro',/get_lun
> printf,lun,'function file,vars'
> printf,lun,'return,vars(0)+vars(1)*0.2'
> printf,lun,'end'
> close,lun& free_lun,lun
>
> ;compile the file
> resolve_routine,'file.pro',/is_function
>
> ;call the function:
> result=call_function('file',var1,var2)
>
> ;clean up
> file_delete,'file.pro',/allow_non
>
>
> now the rub here is that you need to know how many variables there are
> to store in that temporary variable. Hopefully, there's some way for
> you to loop over that.

```

You can't compile a file from the VM, so I think you are stuck actually parsing the expression.

```

> The alternative way, would be to parse the command using the usual
> order of operations rules. Then process that parsed things using pre-
> defined functions for the four arithmetic operations. Such as:
>
> t=multiply(v1,v2)
> t=divide(v1,v2)
>
> etc. WHere they have the obvious "innards". Of course, this can get
> very tedious, but is very useful. If you go down this path, I
> recommend using the byte typecasting to find the major operators:
> t=byte(cmd).
>
>
> Good Luck
> Russell
>
> PS, the reason this is so complicated, is that they don't want you
> doing this.

```

Mike

--

Michael Galloy  
www.michaelgalloy.com  
Modern IDL, A Guide to Learning IDL: <http://modernidl.idldev.com>  
Research Mathematician  
Tech-X Corporation

---