Subject: Re: Application programming--missing features
Posted by jdlb on Mon, 19 Apr 1993 21:47:12 GMT
View Forum Message <> Reply to Message

hadfield_m@kosmos.wcc.govt.nz writes:
> As far as I am aware, there are no restrictions on the number of
> characters in a string fed to EXECUTE.

Actually, there is a miserly 131-character limit, according to the manual. One of my colleagues here (Hi, Tom!) cleverly works around the limit by writing a temporary procedure which includes the command string, using call_procedure to run the thing, and deleting the file.

There is a worse problem with execute: it fills up the code compilation area if you run it repeatedly. I have noticed this with one of my own routines and have just repeated it with the following test:

```
IDL> cmd = 'print,"foo" & print,"bar"'
IDL>print, execute(cmd)  ;works fine 1st few times:
foo
bar
1
IDL> print,execute(cmd)  ;after about 30 repetitions:

print,"foo" & print,"bar"
^
% Program code area full.
0
```

I know the size of the code area can be changed using .SIZE. (I have 64k set aside, which is more than the default.) That's not the issue, because no matter how big the space is made, it's bound to fill up. And once it's full, it seems you can't empty it or do anything useful. (You can issue a .SIZE, which will kill all your variables while it empties the code compilation area.)

Ray Sterner's <sterner@warper.jhuapl.edu> technique of passing keywords to a wrapped routine using a string is clever, and perhaps easiest for the programmer, but difficult for the user because of the non-standard syntax. And Mark Hadfield's <hadfield@wao.greta.cri.nz> comment about variables not being passed because they are merely characters in a string is an important consideration.

Regarding user-defined defaults for customization, I think Ray's suggestion to use environment variables is an excellent one.

--Jeff

% Jeff de la Beaujardiere % jdlb@mamane.ifa.hawaii.edu %
% Institute for Astronomy % 808-956-9843 %
% University of Hawai`i % fax 956-9402 %
