Thank you all for your help, so here is what I am trying to do

I want to write some wrapper functions for some IDL procedure /
functions. I just saw _EXTRA which I might be able to use to do what I
need to do .?

So for example for OPENU, I want to create a wrapper function that will
do some logging and the call the original OPENU:

example:
```
PRO LOG_OPENU, UNIT, FILE, APPEND=APPEND, COMPRESS=COMPRESS,
BUFSIZE=BUFSIZE, DELETE=DELETE, ERROR=ERROR,
F77_UNFORMATTED=F77_UNFORMATTED, GET_LUN=GET_LUN, MORE=MORE,
NOEXPAND_PATH=NOEXPAND_PATH, STDIO=STDIO, SWAP_ENDIAN=SWAP_ENDIAN, $
        SWAP_IF_BIG_ENDIAN=SWAP_IF_BIG_ENDIAN,
SWAP_IF_LITTLE_ENDIAN=SWAP_IF_LITTLE_ENDIAN, VAX_FLOAT=VAX_FLOAT,
WIDTH=WIDTH, XDR=XDR, RAWIO=RAWIO
   print, FILE
   print, SWAP_IF_BIG_ENDIAN
   ;print etc...
   ;Now here would like to call the original method but I have to call
it only with the named arguments that are set
   OPENU, ...?
```

So, either I could have a LOOOONG list of if or case statements or I
maybe I could use _EXTRA?
Or:
I would need to build the method call dynamically:
1) Is there a hack around this so that I could use call_function,
call_procedure or call_method dynamically?

2) Or at least build an array of the arguments so that I could do that
alternatively? Like in Perl where you can pass in an array of
arguments... hmmm, that might not work for name arguments though...

3) Or a way to use _EXTRA

thanks,

dometz

Maarten wrote:
> Dometz wrote:

>> PS: If I call a function with EXECUTE, is there a way to get the return
>> value?
>
> execute, well, executes a full statement. If you set the string to
> "retVal = function(...)", then after the call the local variable retVal
> will hold the result of the function call.
>
> Maarten
>
> (PS, there are many good reasons to try to avoid execute, call_xxx
> offer more flexibility, and far more speed. Perhaps if you give more
> details of what you want to achieve, and why it is hard to use the
> call_xxx variants, we can offer suggestions.)