Subject: Re: objects and call external
Posted by Gert Van de Wouwer on Wed, 13 Feb 2002 13:18:47 GMT
View Forum Message <> Reply to Message

all right, so passing is self.hcom is pass by value; passing hcomm is pass
by reference. Is there a way to pass self.hcomm by reference; i.e. the
c-analgo would be &(self.hcomm)...?

"Mark Rivers" <rivers@cars.uchicago.edu> wrote in message
news:RLla8.194$x4.4337@news.uchicago.edu...
>
> Gert <gert.van.de.wouwer@NO_SPAMpandora.be> wrote in message
> news:SJea8.136468$rt4.12914@afrodite.telenet-ops.be...
>> hi,
>>
>> I try to use a call_external in an object method like this:
>
>
>> pro MCP2000__DEFINE
>> struct = {MCP2000, hComm: 0l, status: 0l}
>> end
>
>> function MCP2000::Init
>> self.hComm = 0l
>> return, 1
>> end
>
>> function MCP2000::InitPort
>>
>
self.status=call_external('D:\Cpp\SerCommDll\Debug\SerCommDl l.dll','InitPort
> Dll',$
>> /PORTABLE,'COM1',self.hComm ,/UNLOAD)
>> return, self.status
>> end
>>
>> the idea is here that self.hComm contains a valid handle, but it
doesnt -
> it stays zero.
>
> Your problem has nothing to do with CALL_EXTERNAL or the fact that you are
> using an object method. The problem is that when you pass self.hComm (to
> any routine) IDL views that as an "expression" and passes a copy of
> self.hComm, not the address of self.hComm. Thus you cannot modify
> self.hComm in the called routine. It is analogous to C passing integers
by

> value - the called routine can write into the function parameter, but the
> calling routine does not see the resulting change. Your second way of doing
> it is correct, pass "temp" and copy "temp" to self.hComm on the return.
>
> Mark Rivers
>
>
>
