Subject: Re: Null terminated strings
Posted by thompson on Tue, 08 Jan 2002 17:17:13 GMT
View Forum Message <> Reply to Message

James Kuyper <kuyper@gscmail.gsfc.nasa.gov> writes:

> Craig Markwardt wrote:

>> James Kuyper <kuyper@gscmail.gsfc.nasa.gov> writes:
>>
>>
>>> I'm reading a string-valued file attribute from an HDF file that was
>>> created using C code. As seems quite reasonable for C programs, the
>>> attribute was written with a length that includes a terminating null
>>> character. When I read it in using IDL, that null character got included
>>> as well. This causes a number of bizarre effects, most notably:
>>>
>>> IDL> print,date
>>> 2001-10-07
>>> IDL> print,date+'T12:00:00'
>>> 2001-10-07'T12:00:0
>>>
>>> I can handle this particular case by using strmid(date,0,10), but in
>>> general a file attribute might contain multiple null-delimited strings,
>>> of unknown length. Is there an efficient way of converting such a string
>>> into an IDL string array?
>>
>>
>> What happens when you swizzle it through a STRING-BYTE-STRING
>> transformation?
>>
>> I.e.,
>>
>> date = string(byte(date))
>>
>> I believe that STRING will ignore any trailing 0-bytes, hence this may
>> solve your problem exactly, at the expense of some extra CPU.

> Thanks - that worked. It only solves the single-string case, but that's
> the case I am currently facing. It saves me the trouble of figuring out
> how long the string is, and it does the right thing, whether or not the
> string is null-terminated.

> I'm still wonder how to best convert a null-delimited list of strings
> into an IDL string array (it's just curiousity, I don't have any
> immediate need for that ability). My best solution so far is to convert
> it to a byte array, find the null delimiting characters with where(),
> and then write a loop to convert each subarray into a seperate IDL

> string. This should work, but I'm always suspicious of the efficiency of
> any solution for an IDL problem that involves an explicit loop.

As far as I can determine, that should work equally as well with arrays as with strings. For example,

```
IDL> test = ['This','is','a','test']
IDL> btest=byte(test)
IDL> print,btest
  84 104 105 115
 105 115   0   0
  97   0   0   0
 116 101 115 116
IDL> stest = string(btest)
IDL> help,stest
STEST           STRING    = Array[4]
IDL> print,strlen(stest)
          4           2           1           4
IDL> print,stest
This is a test
```

You shouldn't have to use a loop.

Bill Thompson
