Subject: Re: Proper use of assoc
Posted by Jeff Guerber on Sat, 14 Dec 2002 00:50:20 GMT
View Forum Message <> Reply to Message

I think David identified the cause of your problem; something that _does_ work, however, is to make the associated variable the target of a pointer:

```
pro fileSet::createAssociation, filename,samples
   openu, lun, filename, /GETLUN, ERROR = err
   IF (err NE 0) then PRINTF, -2, !ERROR_STATE.MSG
   self.sonarDatap =  ptr_new( assoc(lun,uintarr(samples)) )
   self.lun = lun
   return
end

;; An example of using self.sonarDatap:
pro fileSet::dumpFile
   n = 0L
   while not eof(self.lun) do begin
      print, n, (*self.sonarDatap)[n]
      n = n + 1L
   endwhile
   return
end

pro fileSet__define
   void = { lun:0L, $
     sonarDatap: ptr_new() }  ; Plus whatever else goes in your object
   return
end
```

Note that the parentheses in "(*self.sonardatap)[n]" are required; otherwise IDL will try to take what the Nth element of self.sonardatap (which isn't an array) points to, instead of taking the Nth element of what self.sonardatap points to. Ie., *self.sonardatap[n] is equivalent to *(self.sonardatap[n]). (The prefix syntax and precedence for pointer dereferencing appears to have come straight from C. No less an authority on C than Dennis Ritchie has indicated that in retrospect, this may have been a mistake! (http://cm.bell-labs.com/who/dmr/chist.pdf, in the section "Critique".))

Also note that EOF() still works with associated variables, or rather their unit numbers. Oh, and it's a good idea to open files with /GETLUN in cases like this, in case you someday want to have more than one fileSet object at the same time. And don't forget a cleanup method that does a "freelun, self.lun" (which also closes the file) and frees the pointer!

I think probably, an associated array basically boils down to a function call that masquerades as a array, and that's why it has so many peculiarities. (Here's another: n_elements() always returns 1.) Hope this helps,

Jeff Guerber

On Fri, 13 Dec 2002, David Fanning wrote:

> Arthur (abbotta@annapolis.nscc.ns.ca) writes:
>
>> Hi. I'm having a problem using associated i/o. I have a series of
>> files that contain arrays of uints. I'm trying to use assoc to be
>> able to access the arrays, but I so far have had no luck in getting
>> it to work.
>>
>> The error that I'm getting is: "File expression not allowed in this
>> context:<UINT FILE>". I get this error when I try to execute the
>> assoc statement. I've checked for an error when the file is opened,
>> but have detected none.
>>
>> Can anybody point out what I'm doing wrong?
>>
>> pro fileSet::createAssociation, filename,samples
>>
>> openu,1,filename, ERROR = err
>> IF (err NE 0) then PRINTF, -2, !ERROR_STATE.MSG
>> self.sonarData = assoc(1,uintarr(samples))
>> end
>
> The problem here is that whatever it is that ASSOC
> returns, can't be stored in whatever type field
> self.sonarData is. :-(
>
> The return variable from ASSOC is a funny thing,
> really. Not any type of IDL variable, as far as
> I can see. Which pretty much eliminates it being
> stored in any kind of a structure.
>
> What I have done before is passed around the logical
> unit number I want to have associated with the filename
> (and maybe the filename itself) so that I can always
> create a LOCAL associated variable in the program module
> where I need it. I think that is the best you can do.
>
> Cheers,
>
> David