Subject: Creating Variables in Programs
Posted by p.phillips on Fri, 10 Jul 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Does anyone know of a way to create a new array under program control, ie
create a string and use that string to make an array. As far as I can see
this is impossible in IDL?

--
Perry Phillips   p.phillips@mail.utexas.edu

Subject: Re: Creating Variables in Programs
Posted by davidf on Tue, 14 Jul 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Riemar Bauer (r.bauer@fz-juelich.de) writes:

```
> An other way to get an undefined
> variabel is b=n_elements(a)
> Yes b is well defined by 0 but a is defined as undefined.
>
> help
> % At  $MAIN$
> B           LONG      =          0
> A           UNDEFINED = <Undefined>
```

Uh, this *only* happens if A is undefined to start with
and is perfectly normal behavior. In fact, it is the only
way to know if keywords are undefined in an IDL procedure or
function:

```
  IF N_Elements(keyword) EQ 0 THEN keyword = 5
```

This construction NEVER makes A undefined (or there is
something seriously wrong with your version of IDL):

```
  A = 5
  B = N_Elements(A)
  HELP, A, B

  A  INT  =  5
  B  LONG =  1
```

Cheers,

David
--

David Fanning, Ph.D.
Fanning Software Consulting
E-Mail: davidf@dfanning.com
Phone: 970-221-0438
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: Creating Variables in Programs
Posted by R. Bauer on Tue, 14 Jul 1998 07:00:00 GMT

View Forum Message <> Reply to Message

Martin Schultz wrote:

```
> Craig Markwardt wrote:
>>
>>> IDL> r=execute('a=fltarr(200)')
>>> IDL> help,a
>>> A               FLOAT     = Array[200]
>>>
>>
>> There is a "gotcha."  in the case of a compiled procedure, the
>> variable "a" must have already been defined.  The following is usually
>> sufficient:
>> [...]
>
> Huh? Here is a little program:
>
> -------------------------------------
> pro testexec,name
>
>    r=execute(name+'=findgen(10)')
>
>    print,r
>    print,b
> return
> end
> -------------------------------------
>
> Of course, you have to call it as testexec,'b' in order to have it work
> properly ;-), but it demonstrates that you don't have to have your
> variable initialized!!
>
> But I don't really see the point of the original question: why the h...
> do you want to do this? To my knowledge, creating variables only makes
> sense if you know what to do with them afterwards - and in order to do
> something with them, you must know their name beforehand. If you want to
> export your newly created variables to the main program or some other
> procedure, you would have to proceed completely different. I would
```

> create a structure with
>   template = { name:'', pvalue:ptr_new() }
> (or an array of these structures with replicate(...))
>
> then manipulatge the string 'name=expression' to 'tmp=expression', store
> the 'name' field in the name tag of the structure and
> pvalue=ptr_new(tmp) will save the value.
>
> This would act as a container (sounds awfully like OOP doesn't it ?),
> and you would have to do a lot of type and error checking in any routine
> that uses the information in this structure (array). Note, that IDL
> itself would not "know" anything about your variables - but, as I said,
> it doesn't make sense if it had to.
>
> ... and don't forget to clean up your heap once a while...
>

Hi Martin,

that's not totally correct.
idl knows a lot of your variables which are defined or defined as undefined
(a=n_elements(b))

print,routine_names(/variables)

for more look in the by now obsolete routine gethelp

I am using this mechanism to create a dynamical structure where are nearby
100 names with definitions (mostly descriptions for datasets like:
experiment,PI_name, param_long_name, param_units ...) are defined. And all
of them which are defined in a program will go into a structure.
In the program I have only to define param_units='K' and later on it will
be a tag name in a structure. All whats in the structure is could be
written to somewhere e.g. netCDF.

Reimar

--
R.Bauer

Institut fuer Stratosphaerische Chemie (ICG-1)
Forschungszentrum Juelich
email: R.Bauer@fz-juelich.de

Subject: Re: Creating Variables in Programs

Posted by R. Bauer on Tue, 14 Jul 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Craig Markwardt wrote:

```
> Martin Schultz <mgs@io.harvard.edu> writes:
>>
>> Perry Phillips wrote:
>>>
>>> Does anyone know of a way to create a new array under program control, ie
>>> create a string and use that string to make an array. As far as I can see
>>> this is impossible in IDL?
>>>
>>> --
>>> Perry Phillips             p.phillips@mail.utexas.edu
>>
>>
>> here's a quick example
>>
>>
>> IDL> r=execute('a=fltarr(200)')
>> IDL> help,a
>> A           FLOAT     = Array[200]
>>
>
> There is a "gotcha."  in the case of a compiled procedure, the
> variable "a" must have already been defined.  The following is usually
> sufficient:
>
>    A = 0
>    ...
>    R = EXECUTE('A=FLTARR(200)')
>
> The IDL internal compiler needs to know that "A" exists before it can
> be assigned to in an EXECUTE statement.  The same applies for
> restoring variables: all the variables in the SAVE file must be
> predefined in the procedure.  Assigning zero to them is fine.
>
>
```

I am not sure if this has changed with idl 5.1.An other way to get an undefined variabel is b=n_elements(a)
Yes b is well defined by 0 but a is defined as undefined.

```
help
% At  $MAIN$
B           LONG      =          0
A           UNDEFINED = <Undefined>
```

Reimar

--
R.Bauer

Institut fuer Stratosphaerische Chemie (ICG-1)
Forschungszentrum Juelich
email: R.Bauer@fz-juelich.de