
Subject: Re: Workaround for lack of foo.([]) capability with structures?

Posted by [Jeremy Bailin](#) on Tue, 25 Jan 2011 12:45:38 GMT

[View Forum Message](#) <> [Reply to Message](#)

On Jan 24, 11:18 pm, Matt Francis <mattjamesfran...@gmail.com> wrote:

> I have a data structure with many tags, and an array of these
> structures holding a bunch of data.
>
> Many of the tags will often not be present or relevant in some
> particular context, and so will be just zeros for the whole array. I
> want to be able to selectively write out to file just those tags that
> contain usefull data.
>
> I have a line like this for the case that I want to write out all the
> tags:
>
> for i=0l,n_elements(foo)-1 do \$
> printf,lunw,foo[i], format=FMT
>
> If however I want to write out not all of the tags, I'm not sure how
> to do this? I can create an appropriate FMT string for the subset of
> tags, and could do something like:
>
> for i=0l,n_elements(foo)-1 do \$
> printf,lunw,foo[i].tag1,foo[i].tag2, format=FMT
>
> where I list just those tags I want written out. This hard codes what
> tags to write though. Since there are many possible combinations of
> which ones I want written out, I'd need dozens of IF/THEN lines like
>
> if (want tags 1 and 2) then begin
> for i=0l,n_elements(foo)-1 do \$
> printf,lunw,foo[i].tag1,foo[i].tag2, format=FMT'
> endif else if (want tags 1 and 3) then begin
> for i=0l,n_elements(foo)-1 do \$
> printf,lunw,foo[i].tag1,foo[i].tag3, format=FMT'
> endelse
>
> This is clearly not the solution. I can easily create an array
> indicating the tags I want written out and would love to be able to
> simply use the command:
>
> for i=0l,n_elements(foo)-1 do \$
> printf,lunw,foo[i].(indx), format=FMT
>
> but IDL (at least my V7.1) does not allow this kind of indexing. Tags
> can only be directly indexed, not via arrays of indices.
>

```
> I could do something like
>
> for i=0l,n_elements(foo)-1 do begin
>   for j=0,ntags-1 do begin
>     printf,lunw,foo[i].(indx[j])
>   endfor
> endfor
>
> however this creates a newline for each printf statement, and I need
> all the data for each array element on one line.
>
> Any ideas?
```

How about using something like that last case to construct a string,
and then print it out at the end of the outer for loop

```
for i=0l,n-1 do begin
  str=""
  for j=0,ntags-1 do begin
    prints, str, foo[i].(indx[j])
  endfor
  printf, lunw, str
endfor
```

-Jeremy.

Subject: Re: Workaround for lack of foo.([]) capability with structures?

Posted by [H. Evans](#) on Wed, 26 Jan 2011 16:53:15 GMT

[View Forum Message](#) <> [Reply to Message](#)

Or add a '\$' to the end of the fmt, e.g.

```
Printf,lun,form='(a,$)',v
```

This suppresses the newline at the end of the print.

On Jan 25, 1:45 pm, Jeremy Bailin <astroco...@gmail.com> wrote:

```
> On Jan 24, 11:18 pm, Matt Francis <mattjamesfran...@gmail.com> wrote:
```

```
>
>
>
>
>
```

```
>> I have a data structure with many tags, and an array of these
>> structures holding a bunch of data.
```

```
>
```

```
>> Many of the tags will often not be present or relevant in some
>> particular context, and so will be just zeros for the whole array. I
```

```

>> want to be able to selectively write out to file just those tags that
>> contain useful data.
>
>> I have a line like this for the case that I want to write out all the
>> tags:
>
>> for i=0,n_elements(foo)-1 do $
>>   printf,lunw,foo[i], format=FMT
>
>> If however I want to write out not all of the tags, I'm not sure how
>> to do this? I can create an appropriate FMT string for the subset of
>> tags, and could do something like:
>
>> for i=0,n_elements(foo)-1 do $
>>   printf,lunw,foo[i].tag1,foo[i].tag2, format=FMT
>
>> where I list just those tags I want written out. This hard codes what
>> tags to write though. Since there are many possible combinations of
>> which ones I want written out, I'd need dozens of IF/THEN lines like
>
>> if (want tags 1 and 2) then begin
>>   for i=0,n_elements(foo)-1 do $
>>     printf,lunw,foo[i].tag1,foo[i].tag2, format=FMT'
>>   endif else if (want tags 1 and 3 ) then begin
>>     for i=0,n_elements(foo)-1 do $
>>       printf,lunw,foo[i].tag1,foo[i].tag3, format=FMT'
>>     endelse
>
>> This is clearly not the solution. I can easily create an array
>> indicating the tags I want written out and would love to be able to
>> simply use the command:
>
>> for i=0,n_elements(foo)-1 do $
>>   printf,lunw,foo[i].(indx), format=FMT
>
>> but IDL (at least my V7.1) does not allow this kind of indexing. Tags
>> can only be directly indexed, not via arrays of indices.
>
>> I could do something like
>
>> for i=0,n_elements(foo)-1 do begin
>>   for j=0,ntags-1 do begin
>>     printf,lunw,foo[i].(indx[j])
>>   endfor
>> endfor
>
>> however this creates a newline for each printf statement, and I need
>> all the data for each array element on one line.

```

```
>
>> Any ideas?
>
> How about using something like that last case to construct a string,
> and then print it out at the end of the outer for loop
>
> for i=0!,n-1 do begin
>   str=""
>   for j=0,ntags-1 do begin
>     prints, str, foo[i].(indx[j])
>   endfor
>   printf, lunw, str
> endfor
>
> -Jeremy.
```
