Subject: "foreach" loops in IDL
Posted by alaniwiusenet on Fri, 16 Jan 2009 18:30:24 GMT
View Forum Message <> Reply to Message

In case anyone's interested, I've produced a little hack that lets you
have "foreach" loops in IDL, i.e. loop over the contents of an array.
It would be nice if the syntax were to let you have something like:

```
for myval in myarray do begin
    print, myval
endfor
```

Unfortunately it doesn't, unless I've missed something somewhere. And
it's just a little fiddly to have to loop over an array index every
time, e.g.:

```
for i=0, n_elements(myarray) - 1 do begin
   myval = myarray[i]
   print, myval
endfor
```

The hack lets you use the following syntax:

```
foreach, 'myval', myarray
@do
    print, myval
@done
```

and nested loops are supported.

If anyone is interested in having this, grab this a (tiny) download,
that contains a few short files to put somewhere in your IDL_PATH:

http://home.badc.rl.ac.uk/iwi/idl-foreach.tar.gz

Regards,
Alan

P.S. Please note that I do not read this mailbox. Do a web search for
"Alan Iwi" if you want my email address.

Subject: Re: "foreach" loops in IDL
Posted by rtk on Tue, 20 Jan 2009 22:54:39 GMT
View Forum Message <> Reply to Message

On Jan 20, 2:11 pm, JD Smith <jdtsmith.nos...@yahoo.com> wrote:
> foreach elem in x

> print, my_function(elem)
> end
>
> When treating a vector as a list, requiring an extra loop variable is
> pure syntactic overhead. Not to mention that the loop variable could
> overflow, could get changed in the body of the loop or, most commonly,
> risks nested sub-loops accidentally re-using the same loop variable.
> None of these happens with a foreach construct.

Exactly! Of course, at this point, we are talking Python:

```
for elem in x:
   print my_function(elem)
```

Lists are a natural extension/complement to IDL's arrays. Then, once you have lists, you will naturally want higher-order functions to operate on those lists. Lists would add great power to IDL but perhaps if people are not familiar with lists they will not appreciate what lists have to offer. To prove my point, look at what happened with Python. It supports lists out of the box but folks added powerful array processing via the numpy package. Well, IDL is just the opposite. It has powerful array processing already, now it just needs lists.

The package I referenced above has a list DLM that is quite useful on its own. It supports all IDL data types except structures. A truly integrated version would naturally add helpful syntax and support all data types.

Ron
(Disclaimer: while I do work for ITT VIS, these opinions are my own and should not be thought of as those of ITT VIS...)

Subject: Re: "foreach" loops in IDL
Posted by Vince Hradil on Tue, 20 Jan 2009 22:57:46 GMT
View Forum Message <> Reply to Message

On Jan 20, 3:11 pm, JD Smith <jdtsmith.nos...@yahoo.com> wrote:
> On Jan 16, 3:43 pm, Vince Hradil <vincehra...@gmail.com> wrote:
>
>> I would have to agree with Paolo here. The for-loop syntax is simple
>> and clear. The @do @done really obfuscates the code.
>
> That's because it's an unsupported add-on. If IDL included foreach at
> the language level, which would be clearer:
>
> for i=0,n_elements(x)-1 do begin

> elem=x[i]
> print, my_function(elem)
> end
>
> or
>
> foreach elem in x
> print, my_function(elem)
> end
>
> When treating a vector as a list, requiring an extra loop variable is
> pure syntactic overhead. Not to mention that the loop variable could
> overflow, could get changed in the body of the loop or, most commonly,
> risks nested sub-loops accidentally re-using the same loop variable.
> None of these happens with a foreach construct.
>
> JD

Oh, I agree - a foreach loop would be great to have in IDL. My only point was that add-on is confusing to read, and I would prefer to write more portable code that uses as much built in functionality as possible.

Subject: Re: "foreach" loops in IDL
Posted by Giorgio on Fri, 23 Jan 2009 23:39:42 GMT
View Forum Message <> Reply to Message

Is it possible to have the DLM built for a 64-bit linux machine?

Thanks
