Subject: Re: "foreach" loops in IDL
Posted by rtk on Fri, 16 Jan 2009 18:54:02 GMT
View Forum Message <> Reply to Message

On Jan 16, 11:30 am, alaniwiuse...@googlemail.com wrote:
> In case anyone's interested, I've produced a little hack that lets you
> have "foreach" loops in IDL

Neat hack!

Along the same lines, I have a set of DLMs here that add things like foreach, map, filter, reduce, etc. to IDL along with lambda functions and lists:

http://www.ittvis.com/info/hof/

For example, with the foreach DLM, your example becomes:

foreach, lambda('x: print, x', /P), myarray

There is also a pure IDL version for functions called 'each()' which is quite useful.

Map, filter and reduce work pretty much as they do in Python (with extensions). There are also compose, apply, cart, accumulate, and group higher-order functions. With cart() it is possible to do the equivalent of Python list comprehensions.

If you try it and have any questions let me know.

Ron
oneelkruns@hotmail.com

Subject: Re: "foreach" loops in IDL
Posted by pgrigis on Fri, 16 Jan 2009 19:37:07 GMT
View Forum Message <> Reply to Message

My opinion is that something like that make the code more difficult to understand and prevent utilization in two different programs running at the same time in the same session because of the common blocks.

Therefore, I don't like the idea very much.

Ciao,
Paolo

alaniwiuse...@googlemail.com wrote:
> 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 Fri, 16 Jan 2009 20:18:17 GMT
View Forum Message <> Reply to Message

On Jan 16, 12:37 pm, Paolo <pgri...@gmail.com> wrote:
> My opinion is that something like that make the
> code more difficult to understand and prevent
> utilization in two different programs running at the

> same time in the same session because of the
> common blocks.

It is unclear which set of extensions you are referring to, but if you mean the ones I mentioned I encourage you to take a second look at lambda.pro. There will be no problem between programs because of the common block. Also, the extensions are meant mostly for command line use.

As for being hard to read and understand, that is just a matter of experience and opinion. Functional languages do pretty well with constructs like these and vastly more sophisticated ones.

Lastly, as always, if you don't like something, don't use it :)

Ron

Subject: Re: "foreach" loops in IDL
Posted by pgrigis on Fri, 16 Jan 2009 20:34:16 GMT
View Forum Message <> Reply to Message

I was referring to foreach.pro , not to what you posted.
Sorry for the confusion.

I guess that I have the same loathing for common blocks that David has for heap_gc ... I guess anybody is entitled his quirks ;-)

Ciao,
Paolo

rtk wrote:
> On Jan 16, 12:37 pm, Paolo <pgri...@gmail.com> wrote:
>> My opinion is that something like that make the
>> code more difficult to understand and prevent
>> utilization in two different programs running at the
>> same time in the same session because of the
>> common blocks.
>
> It is unclear which set of extensions you are referring to, but if you
> mean the ones I mentioned I encourage you to take a second look at
> lambda.pro. There will be no problem between programs because of the
> common block. Also, the extensions are meant mostly for command line
> use.
>
> As for being hard to read and understand, that is just a matter of
> experience and opinion. Functional languages do pretty well with

> constructs like these and vastly more sophisticated ones.
>
> Lastly, as always, if you don't like something, don't use it :)
>
> Ron

Subject: Re: "foreach" loops in IDL
Posted by Vince Hradil on Fri, 16 Jan 2009 20:43:00 GMT
View Forum Message <> Reply to Message

On Jan 16, 3:34 pm, Paolo <pgri...@gmail.com> wrote:
> I was referring to foreach.pro , not to what you posted.
> Sorry for the confusion.
>
> I guess that I have the same loathing for common blocks that
> David has for heap_gc ... I guess anybody is entitled his
> quirks ;-)
>
> Ciao,
> Paolo
>
>
>
> rtk wrote:
>> On Jan 16, 12:37 pm, Paolo <pgri...@gmail.com> wrote:
>>> My opinion is that something like that make the
>>> code more difficult to understand and prevent
>>> utilization in two different programs running at the
>>> same time in the same session because of the
>>> common blocks.
>
>> It is unclear which set of extensions you are referring to, but if you
>> mean the ones I mentioned I encourage you to take a second look at
>> lambda.pro. There will be no problem between programs because of the
>> common block. Also, the extensions are meant mostly for command line
>> use.
>
>> As for being hard to read and understand, that is just a matter of
>> experience and opinion. Functional languages do pretty well with
>> constructs like these and vastly more sophisticated ones.
>
>> Lastly, as always, if you don't like something, don't use it :)
>
>> Ron- Hide quoted text -
>
> - Show quoted text -

I would have to agree with Paolo here. The for-loop syntax is simple and clear. The @do @done really obfuscates the code.

Subject: Re: "foreach" loops in IDL
Posted by rtk on Fri, 16 Jan 2009 21:07:54 GMT
View Forum Message <> Reply to Message

On Jan 16, 1:34 pm, Paolo <pgri...@gmail.com> wrote:
> I was referring to foreach.pro , not to what you posted.
> Sorry for the confusion.
>
> I guess that I have the same loathing for common blocks that
> David has for heap_gc ... I guess anybody is entitled his
> quirks ;-)

:) I certainly have plenty of my own!

I don't like common blocks much, either, really. I have used them to act as class level data for some classes I've written (ie, data accessible to all instances of that class) but you could replace the common block there and in my lambda.pro with a top level variable if necessary. The whole "lambda()" bit is a hack anyway to give the illusion of something IDL doesn't really have.

Ron

Subject: Re: "foreach" loops in IDL
Posted by R.Bauer on Mon, 19 Jan 2009 10:31:35 GMT
View Forum Message <> Reply to Message

rtk schrieb:
> On Jan 16, 11:30 am, alaniwiuse...@googlemail.com wrote:
>> In case anyone's interested, I've produced a little hack that lets you
>> have "foreach" loops in IDL
>
> Neat hack!
>
> Along the same lines, I have a set of DLMs here that add things like
> foreach, map, filter, reduce, etc. to IDL along with lambda functions
> and lists:
>
> http://www.ittvis.com/info/hof/
>
> For example, with the foreach DLM, your example becomes:
>

> foreach, lambda('x: print, x', /P), myarray
>
> There is also a pure IDL version for functions called 'each()' which
> is quite useful.
>
> Map, filter and reduce work pretty much as they do in Python (with
> extensions). There are also compose, apply, cart, accumulate, and
> group higher-order functions. With cart() it is possible to do the
> equivalent of Python list comprehensions.
>
> If you try it and have any questions let me know.
>
> Ron
> oneelkruns@hotmail.com
>

Some more hacks and we can call it python

cheers
Reimar

/me lives in both worlds already

Subject: Re: "foreach" loops in IDL
Posted by JD Smith on Tue, 20 Jan 2009 21:11:12 GMT
View Forum Message <> Reply to Message

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

Subject: Re: "foreach" loops in IDL
Posted by alaniwiusenet on Sat, 31 Jan 2009 12:20:24 GMT
View Forum Message <> Reply to Message

On 20 Jan, 21:11, JD Smith <jdtsmith.nos...@yahoo.com> wrote:
>
> That's because it's an unsupported add-on. If IDL included foreach at
> the language level, which would be clearer: [...]

Exactly.

Of course the hack is a bit ugly. It is more intended for use in
quick-and-dirty scripts than anything you intend to distribute etc.

My original post was also partly intended as a way to express a desire
for inclusion of "foreach" at the language level. Anyone at ITTVIS
reading this? If so, please also allow empty lists while you're at
it. Thanks. :-)

Alan

====
Again please note that I do not read this mailbox. Do a web search
for "Alan Iwi" if you want my email address.
