Subject: Self-compiling procedures
Posted by maye on Sat, 16 Feb 2008 18:47:54 GMT
View Forum Message <> Reply to Message

Hi!
In terms of self-compiling I'm a beginner, so please excuse my
ignorance if this is written somewhere in the manual, I couldn't find
it easily at least:
So far I put every procedure I consider to be useful enough for more
than one task in its extra file, so that if it's called, IDL compiles
it automatically, having the procedure and file name identical.
But after a while, the number of files I have keeps growing.
I wonder if there's another elegant way to keep things auto-compilable
without me having to compile some kind of library before I start
working on a task using my "library" of useful procedures.
Am I missing something simple?
Thanks for any hints and a nice weekend!
Best regards,
Michael

Subject: Re: Self-compiling procedures
Posted by Maarten[1] on Tue, 19 Feb 2008 09:56:51 GMT
View Forum Message <> Reply to Message

On Feb 19, 9:29 am, Michael Aye <kmichael....@googlemail.com> wrote:
> On Feb 18, 9:55 am, Maarten <maarten.sn...@knmi.nl> wrote:
>> If the functions are truly related, you may want to investigate the
>> use of object programming. The IDL manual on the subject is quite
>> horrible - it assumes you want to use their object graphics and write
>> user interfaces in IDL (yuck). But as far as grouping related
>> function, it is a powerful concept. Be prepared to dive into pointers
>> though.
>
> Yes, saw the object stuff already, but wondered, as you said, if
> that's only there for object graphics or if there's more use to it. So
> it's a good point.

There are a few analysis objects, and yes: objects are useful outside
of the graphics and UI context. Since the objects are bolted onto the
structures, you also inherit (no pun intended) the limitation of named
structures: arrays have to be pointers, or you won't be able to resize
them.

> Which reminds me of another question since IDL 7.0: What's ITT's idea
> now for graphical UI's?
> And are there actually ways to do the UI e.g. with QT Designer or with
> WXWindowx/WxPython and put it on top of IDL routines?

IDL has its own UI toolkit (probably based on something more or less
native). I /never/ use IDL for user interfaces, the 'I' in IDL is
pointless to me at this moment, and I don't think that will change. I
would say: just don't bother with IDL user interfaces, especially if
you already know Python and WxPython. Those are much cleaner, more
modern, and far less frustrating. And some algorithms are just plain
easier to write and read back - if you need to obtain a fair speed.
'for' loops don't kill Python like they do IDL.

Maarten

Subject: Re: Self-compiling procedures

Posted by mmiller3 on Tue, 19 Feb 2008 16:02:21 GMT

View Forum Message <> Reply to Message

>>>> > "Maarten" == Maarten <maarten.sneep@knmi.nl> writes:

> I /never/ use IDL for user interfaces, the 'I' in IDL is
> pointless to me at this moment, and I don't think that will
> change. I would say: just don't bother with IDL user
> interfaces, especially if you already know Python and
> WxPython. Those are much cleaner, more modern, and far less
> frustrating. And some algorithms are just plain easier to
> write and read back - if you need to obtain a fair speed.
> 'for' loops don't kill Python like they do IDL.

Do you mean that you never make widget-based user interfaces for
IDL? Or that you use WxPython to make widget-based user
interfaces for IDL? If the later, would you share some tips?

Mike

Subject: Re: Self-compiling procedures

Posted by Maarten[1] on Tue, 19 Feb 2008 16:20:24 GMT

View Forum Message <> Reply to Message

On Feb 19, 4:02 pm, mmill...@iupui.edu (Michael A. Miller) wrote:
>>>> >> "Maarten" == Maarten <maarten.sn...@knmi.nl> writes:
>
>> I /never/ use IDL for user interfaces, the 'I' in IDL is
>> pointless to me at this moment, and I don't think that will
>> change. I would say: just don't bother with IDL user
>> interfaces, especially if you already know Python and
>> WxPython. Those are much cleaner, more modern, and far less

>> frustrating. And some algorithms are just plain easier to
>> write and read back - if you need to obtain a fair speed.
>> 'for' loops don't kill Python like they do IDL.
>
> Do you mean that you never make widget-based user interfaces for
> IDL?

That is what I meant.

> Or that you use WxPython to make widget-based user
> interfaces for IDL? If the later, would you share some tips?

No, and I would not attempt that (if I were you). Python will do
nicely: numpy, scipy, and pytables serve my needs quite nicely on that
side.

Maarten

Subject: Re: Self-compiling procedures
Posted by mmiller3 on Tue, 19 Feb 2008 17:00:11 GMT

View Forum Message <> Reply to Message

>>>> > "Maarten" == Maarten <maarten.sneep@knmi.nl> writes:

> No, and I would not attempt that (if I were you). Python
> will do nicely: numpy, scipy, and pytables serve my needs
> quite nicely on that side.

I have written lots of things using IDL's widget library. It is
limited, but I claim that those limitations free me from having
to make too many complicated design choices :-). I've done a lot
with python as well, but have never found anything that is as
convenient as IDL for image processing and interaction.

Mike

Subject: Re: Self-compiling procedures
Posted by David Fanning on Tue, 19 Feb 2008 17:26:49 GMT

View Forum Message <> Reply to Message

Michael A. Miller writes:

> I have written lots of things using IDL's widget library. It is
> limited, but I claim that those limitations free me from having
> to make too many complicated design choices :-). I've done a lot

> with python as well, but have never found anything that is as
> convenient as IDL for image processing and interaction.

I guess I've got to get out more, but I find IDL's
widget tools to be powerful and easy to use. I'm
mystified about why people won't use them.

Cheers,

David
--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming (www.dfanning.com)
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: Self-compiling procedures

Posted by Michael Galloy on Tue, 19 Feb 2008 20:09:44 GMT

View Forum Message <> Reply to Message

On Feb 19, 10:26 am, David Fanning <n...@dfanning.com> wrote:
> I guess I've got to get out more, but I find IDL's
> widget tools to be powerful and easy to use. I'm
> mystified about why people won't use them.

I do think that IDL's widgets are easy to use, but they look "out of
place", especially on Linux and Mac. They also don't have a full set
of features that you would expect to find in the native platform
systems, Qt, GTK, etc. For example, displaying rich text would be
useful as well as having more options for tree and tabular data. Many
of these features probably seem like small things, but they add up to
a much better user experience.

But I don't really want ITT VIS spending a lot of their development
time building widgets. I think the best option would be to just have
an interface to some widget toolkit that is already cross-platform
like Qt.

Mike
--
www.michaelgalloy.com
Tech-X Corporation
Software Developer II

Subject: Re: Self-compiling procedures

Posted by David Fanning on Tue, 19 Feb 2008 20:27:49 GMT
View Forum Message <> Reply to Message

mgalloy@gmail.com writes:

> But I don't really want ITT VIS spending a lot of their development
> time building widgets. I think the best option would be to just have
> an interface to some widget toolkit that is already cross-platform
> like Qt.

I don't know anything about Qt, but I'm sure it's terrific.

The problem with relying on 3rd party software, though, is that
when something goes wrong (and it *always* eventually goes wrong),
it is too easy for software developers to point the finger at
one another. I would prefer to know who I am suppose to be
yelling at. For example, are my current problems (IDL seems to
become unresponsive to me at times, usually in the middle
of an IDL course) caused by ITTVIS or by the fine folks doing
the Eclipse development?

I'm afraid some problems (a seemingly random number of spaces
inserted whenever I hit the TAB key) will *never* get fixed. :-(

Cheers,

David
--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming (www.dfanning.com)
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: Self-compiling procedures
Posted by Michael Galloy on Tue, 19 Feb 2008 21:00:34 GMT
View Forum Message <> Reply to Message

On Feb 19, 1:27 pm, David Fanning <n...@dfanning.com> wrote:
> I don't know anything about Qt, but I'm sure it's terrific.
>
> The problem with relying on 3rd party software, though, is that
> when something goes wrong (and it *always* eventually goes wrong),
> it is too easy for software developers to point the finger at
> one another. I would prefer to know who I am suppose to be
> yelling at. For example, are my current problems (IDL seems to
> become unresponsive to me at times, usually in the middle
> of an IDL course) caused by ITTVIS or by the fine folks doing

> the Eclipse development?

It doesn't matter, you get to yell at the ITT VIS people in either case. The fact that they use other software to create their end product doesn't absolve them from guilt if their product doesn't work. And the widget libraries are already third-party software. It's just that IDL creates a common cross-platform interface for differing widget toolkits, then has to deal with all the issues that come up between different GUI systems on different platforms. Why not push that off to (successful) systems that already do that like Qt, GTK, wxWidgets, SWT, etc?

Mike
--
www.michaelgalloy.com
Tech-X Corporation
Software Developer II

Subject: Re: Self-compiling procedures
Posted by mmiller3 on Wed, 20 Feb 2008 15:58:03 GMT
View Forum Message <> Reply to Message

>>>> > "David" == David Fanning <news@dfanning.com> writes:

> I guess I've got to get out more, but I find IDL's widget
> tools to be powerful and easy to use. I'm mystified about
> why people won't use them.

I like IDL's widgets - my comment about design choices was tongue-in-cheek, but I meant it. It is possible to spend hours fiddling with colors and fonts and detailed gui layout with some other toolkits, but with IDL, once I realized it couldn't really be done, my productivity went up. Being able to tell people "no, I can't make that some color other than grey" is a huge time saver! Some interfaces end up looking a bit clunky, and are likely less efficient to use that they could be. I suspect that is why so many commercial programs that are written in IDL use proprietary interface libraries.

The other thing that I've used extensively for GUIs is tkinter (and some plain tcl/tk). The big advantage there, that is missing with IDL, is a very large user base with lots of documentation and examples.

Not counting itools, I've still never found anything that is as handy as IDL for both numerical work and interacting with data, and a lot of that is due the availability of the widgets and direct graphics.

Mike

--
Michael A. Miller   mmiller3@iupui.edu
  Imaging Sciences, Department of Radiology, IU School of Medicine