Subject: Re: compile a routine wich inlude a commun
Posted by David Fanning on Mon, 23 Jan 2006 15:03:48 GMT
View Forum Message <> Reply to Message

Maarten writes:

> I take your word for it, but each time I picked up the RSI introduction
> to object programming in IDL, I threw it away in utter disgust after a
> few pages. With a object programming background in Objective-C (That is
> when I really "got" objects) and later in Python, the objects in IDL
> feel like hacks, and ugly ones at that.

I know how you feel, Maarten. The Denver Broncos got
crushed yesterday in the AFC Football Championship and
most of the folks in my neighborhood think the world is over.
It will get better with time. :-)

And look at the bright side, we don't have to listen to the
Super Bowl hype for the next two weeks!

> Especially python puts the IDL objects to shame.

I'm sure of it.

> Non-resizable arrays
> in objects (or structures for that matter). Aargh! And as a remedy:
> let's introduce pointers. Why do they think I would use a
> scripting/non-compiling language in the first place!

To avoid pointers!? Are you a Luddite? Pointers
are the coolest thing *in* IDL. Global, sticky, variables
that act *exactly* like any other IDL variables. Fantastic!
I think almost everyone would agree it is one thing RSI got
exactly right.

> After half a year of using IDL I really wonder why anyone with half a
> sane mind would use IDL for new projects. Let's enumerate the reasons
> to use IDL:
>
> 1) Legacy code
> 2) Can't afford Matlab
> 3) Struggled to learn it, afraid to throw away that time to learn
> something else
> 4) Popular in the field of interest, so we struggle together, with a
> lot of code available
> 5) Haven't looked too well at other options
> 6) Masochism, believe others when they say that IDL is really powerful

How much is item 3 figuring in your own evaluation of IDL?
In IDL programming courses I teach, I figure as many as a third
of the people in the course won't ever successfully use IDL,
simply because they can't bring themselves to give it a chance.
It's not Fortran, it's not C, it's not Python. The list goes
on and on.

Yes, IDL is a messy language. But have you looked at
programs you wrote 10 years ago? 20? *28* years ago!
Imagine keeping those programs you first punched on card
decks backward compatible. Imagine trying to add new
programming concepts to an old language. Yes, it is messy
and compromised and .... well, you fill in the blank.
I'm sure it is all that.

Yet, there is no better alternative for a number of
users. IDL objects are inelegant, agreed. They are
far from a perfect implementation. But they bring
additional power and capability to a language that
can use them. I've certainly written programs
with them that I didn't think were possible in IDL.
So, I like them despite their obvious limitations.

> The code given there beyond the explicit loop I use here, is so hard to
> read, (and therefore hard to maintain), that I simply put up with the
> slow, but readable, explicit loop. I tried to rewrite one of the faster
> algorithms shown there to a 2D version, but got nowhere. Any
> programming language that forces you to write code that hard to read,
> has fundamental problems, and IMHO should be avoided.

Well, I guess this is my fault. I partly put that
Drizzling page up there *because* it is so hard to
understand. I certainly don't understand it. It's
one of my little Coyote jokes, if you want to know
the truth. But it would be hard to fault the elegance
and simplicity of the small examples to illustrate
the IDL Way page:

http://www.dfanning.com/idl_way/smallexamples.html

I don't know Python, but I would enter the examples
found on that page in any Elegant Programming contest
and expect to have a chance at winning.

> Does this make me popular in this newsgroup? Does it give me a chance
> of getting answers here? *ploink*, I guess.

Oh, I wouldn't worry about it. We are fools enough to answer *anyone's* questions. :-)

> Does it make me feel better? Yes, certainly.

I hope so. And I hope the rest of the week goes
better than today! :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/