Subject: Re: compile a routine wich inlude a commun
Posted by Maarten[1] on Mon, 23 Jan 2006 16:59:46 GMT
View Forum Message <> Reply to Message

David Fanning wrote:
> 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!

I hope that the programming bit will get better over time, whatever the language. As for "missing" any sports hype: I know how you feel, the olympics are coming up, can't wait till they're over ;-)

>> 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.

And I maintain that requiring fixed sized arrays in structures is impractical. Having pointers is no substitute, as far as I'm concerned: way too easy to loose track of, and an array inside a structure needs a different syntax than a pointer to an array. References to data objects are needed as well, but not as the sole means of access to an object.

>> 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:

(note that I say *new* projects).

[snip]

> How much is item 3 [struggle to learn, afraid to throw out invested time]
> figuring in your own evaluation of IDL?

Not at all. Like I said: I'd jump ship the very moment I see the opportunity - it is just that I don't have the time to gather all the items I need on a python install. And the struggle isn't that heavy: apart from objects, where it is plain disgust holding me back, I don't have too much trouble picking things up. I do wonder "what were they thinking" too often though. And yes, 28 years is a long time, but

still, I feel that the direct graphics to object graphics transtition
could have been handled more elegantly.

> 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.

I'm _very_ aware of what IDL is _not_. It is just that I have a hard
time figuring out what it _is_ instead.

> Yes, IDL is a messy language. But have you looked at
> programs you wrote 10 years ago? 20? *28* years ago!

My code cannot be that old. Well, technically it could, but only just.
But yes, coding style changes over time, and generally imrpoves from
leasons learnt. I've seen other programs handle backward compatibility
in other ways, that improved the whole, and left us with slightly less
of a mess.

> 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.

I'm a (La)TeX user, I've done some pretty wild things over there. I
know a thing or two about old code, legacy systems, backward
compatibility and the mess it can create, not to mention programming
languages that think in reverse gear (if you really want: try the
language of bibtex one time. I always get the feeling of needing a
reverse gear on my mind when *reading* the code -- literaly: reading it
from the end to the beginning, and the code actually makes some sense.
And that is just reading.)

Thing is, when starting _now_ you care about this () much about that
legacy. What you see is the mess, and what you miss is what you had
before (and can no longer use because of the change in the job).

> Yet, there is no better alternative for a number of users.

Why? What is unique to IDL that makes it the only option for some
users? Legacy code is an obvious one, but the less obvious ones?

[snip]

> 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.

And yet: what that code does is something that is fairly common, and it is rather silly that it takes code that is _that_ hard. Well, I opt for readability and ease of programming, speed be damned.

> 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.

I'll have a look later on, and see if I can come up with an elegant Python version for some of the samples.

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

Ah, lucky me ;-) On the whole this usenet group is friendly and helpful. I must say that is makes it easier on me to stick with it.

>> Does it make me feel better? Yes, certainly.
>
> I hope so. And I hope the rest of the week goes
> better than today! :-)

Oh, today went rather well, so far at least. Given the local time, that is not too bad.

Maarten
