Subject: Re: The Elements of IDL style
Posted by JD Smith on Mon, 22 Dec 2003 19:52:19 GMT
View Forum Message <> Reply to Message

On Sat, 20 Dec 2003 09:39:43 -0700, David Fanning wrote:

> Michael Wallace writes:
>
>> I hope I'm not opening a can of worms with this question. Maybe I am.
>> If I am, let the fun ensue! ;-)
>>
>> I am curious if there are any community guidelines for coding
>> conventions and style for IDL. I tried googling, but the results I
>> turned up were either "do whatever you want" or just some little blurb
>> about one little itty-bitty facet of the language. I did find a link
>> that looked promising, http://www.rsinc.com/helpfile.asp?wh=14, but
>> alas, I got a lovely empty white web page in return.
>>
>> I'm not just interested in the style of coding things like when to use
>> UPPERCASE or lowercase or MixedCase, but also in what is an appropriate
>> way to name functions and procedures, files and such.
>>
>> The coding standards meetings I've been in over the years have been
>> some of the most heated and hated meetings of all and at the end of the
>> day, when the dust has settled, no real progress was made except that
>> everyone now thinks that everyone else is a stark raving mad idiot. So
>> with that in mind, I don't want to start anything like that. Either
>> standards exist or they don't. If they exist, I'd appreciate someone
>> providing a link to the standards. If not, just tell me so, and I'll
>> make up my own.
>
> Of course there is an IDL Style Guide. It is called the "Coyote Way" and
> I have made it a 16-year quest to get it adopted. So far, only Ben
> Tupper has converted, and then only intermittently. :-(
>
> *Which* style you use is much less important than the fact that you
> *have* style. All decent IDL programmers have style, and it usually
> involves consistent indenting and use of white space, as well as a style
> of capitalization. I've never seen two good programmers use the same
> style, but when you look at a piece of IDL code you know immediately
> whether you are looking at the work of a stylish programmer or not. And
> believe me, if you are, that program is going to be a LOT easier to work
> with and to understand.
>
> A style helps you see how a program works. A good style along with
> decent variable naming conventions allows you to "read" code the way you
> might read a book. (And, of course, a *great* style involves a copious
> amount of program documentation, too.)

To avoid inciting riot, first I'll say it really is a matter of taste. I for one am a big fan of "compact" expression, which means I use much shorter variable names than, e.g., David, but probably with a higher comment/code ratio (especially when doing something hairy with HISTOGRAM).

Consistent indentation is a must: IDLWAVE can help, if you're an Emacs person. Otherwise, pick your style and stick to it. I use IDLWAVE settings:

```
idlwave-main-block-indent 2
idlwave-end-offset -3
idlwave-continuation-indent 3
```

which lines things up like:

```
pro indent_test
  for i=0,2 do begin
     print,i
  endfor
end
```

Continuing line indentation is slightly more subjective, but IDLWAVE has several settings to help you there too (including a new, optional Kernighan-inspired method due in the next release), and can auto-capitalize, space-pad, and enforce many different configurable style conventions. Without it, my code would probably be a mess.

I reserve upper case for all keywords, mixed case for class and method names, and use lower case for almost everything else (variables, function/procedure names, loop statements, etc.). This just saves trips to the shift key for me.

When creating procedures and files, keep in mind the limitations IDL has with routine names: no two routines or classes can have the same name (well, they can, but bad things will result). With most IDL installations having literally ~10,000 or more routines installed, it's important to develop ways to conserve the namespace. One way is to use objects, which segregate all methods under a single class heirarchy. You can also use a method-like convention for normal routines, ala:

```
pro prefix_dosomething
end

function prefix_returnsomething
end
```

This is vital if you plan to distribute your code (or can envision it ever

getting distributed). Many people use their initials in the prefix, RSI uses "IDL", but anything unique to your program will do. A procedure named "calculate" is an example of a poorly named routine.

Other than that, I'll just echo David's comment to pick something and stick to it.

JD
