Subject: Re: Language Documentation
Posted by rob on Sat, 09 Jul 1994 01:43:16 GMT
View Forum Message <> Reply to Message

In article <2vjsic$9p5@ncar.ucar.edu> caron@acd.ucar.edu (John Caron) writes:
>
>> Well, to be correct, a 'true' value is any 'odd' value, i.e. LSB is 1.
>> A 'false' value is any 'even' value, i.e. LSB is 0. The NOT operator
>> does a bitwise NOT, so this works out. WHERE, however, returns indices
>> of all *nonzero* elements, so use -1 because (NOT -1) is 0.
>
> Ive been wondering about this. Could RSI please get this kind of stuff
> into their documents? As a C programmer, I've been used to 0 or not 0 logic.
> Also, you could mention that all your logical operations are bitwise (I think),
> e.g "and" is "&" not "&&".

RSI *does* have this kind of stuff in its documents, e.g., "Definition of True" on page 4-8 of the (Version 3.5) User's Guide, explanation of WHERE on 1-313 of the Reference Guide, etc. I consider their documentation among the better that I've had to deal with, and information is *usually* pretty easy to find.

I would prefer more on-line support, however, e.g., a search facility in the "help GUI" would be very useful (it doesn't exist in old 3.5.1, anyway).

> As a general comment to RSI, your manuals seem written for the casual,
> scientist-type (i.e. non-programmer). Sort of a "dont confuse them" attitude.
> Your language description is woefully vague to the eyes (ears?) of this
> programmer. How about a "programmer's description" of the language?
>
> Comments?

The more computer-science-oriented people here do better with the manuals, whether they're scientists or programmers. I'm not sure what you want specifically for the programmers; I've not found the need for a BNF-like syntax chart "programmer's description," for example.

The biggest problem is when people just don't use the documentation, and of course RSI has no control over that. They do appear receptive to *specific* suggestions for improvements.

The kind of specific suggestions they aren't too receptive to are, as one would guess, changes to the language itself. There are some historically-based "features" that we'll just have to live with.

[Maybe a chapter entitled something like "Significant Differences Between IDL and Some Other Popular Languages" should be added (e.g., Fortran and C)?]

My (biased) view would have the language be more Unix/C/C++ oriented,
e.g., give us aliases, let us use ! for escaping to the shell,
full tcsh command completion and history (recall) would be wonderful,
optional vi-style editing of recalled commands, something like C++
classes/templates/etc. for object-oriented programming would be
excellent in bringing the users into the present and the language
into the future... Well, that's some of my wish-list, anyway. :-)

-Rob
