Subject: Re: simple question (I hope)
Posted by Foldy Lajos on Fri, 30 Mar 2007 19:46:27 GMT
View Forum Message <> Reply to Message

On Fri, 30 Mar 2007, JD Smith wrote:

>> I don't know how IDL is implemented, but I use pass-by-value for temporary
>> variables in FL. Here "pass" means move, not copy ("move semantics"). The
>> original temporary does not exist after entering the called routine, it is
>> undefined. The called routine gets values, not references. It is faster
>> than pass-by-reference, since no de-referencing is needed for these
>> variables.
>
> I just speculate, but given that most IDL variable types use a pointer
> to access their contained data (strings, arrays, etc), from an IDL user
> point of view, this is pass by reference, no matter how you inject the
> thin wrapper around the variable into a routine. That's what I mean by
> by-value vs. by-reference, and I'd guess FL does it the same (?).
>

For scalar numeric values, it is exact pass-by-value. For more complicated
data (strings, arrays) it is pass-by-reference for the internal pointer,
if you like. But there is one big difference: the number of references.
For pass-by-reference, there are more than one valid reference. For
pass-by-value (move), there is always one valid reference. This is very
useful for garbage collecting.

IDL users can not access all the references, but IDL internally can, and
managing data with multiple references is difficult, that's why sometimes
temporary variables are not handled correctly.

regards,
lajos

ps: the "small string optimization" is on my TODO list. After that, small
strings will be passed by value, too.