Subject: Re: -32768
Posted by Jim Pendleton on Thu, 20 Mar 2014 21:11:07 GMT
View Forum Message <> Reply to Message

On Thursday, March 20, 2014 11:01:01 AM UTC-6, Karl wrote:
> On Thursday, March 20, 2014 3:24:59 AM UTC-6, Rob Klooster wrote:
>
>> The reason it does not work, is that you first try to define +32768 and then negate the result. Since +32768 can only be expressed as a long, you end up with a long variable. Using fix is a solution, but you could also use the bitwise not operator, if you feel adventurous:
>
>>
>
>> IDL> a = not 32767
>
>>
>
>> IDL> help, a
>
>>
>
>> A INT = -32768
>
>
>
> And whether this is a "bug" in IDL or not depends on how IDL defines literal constants.
>
>
>
> If an IDL (integral) constant has the form (excuse my abuse of regex):
>
>
>
> [+-][0-9]*
>
>
>
> then one could argue this is be a bug.
>
>
>
> If a constant can be only:
>
>
>
> [0-9]*
>
>

>
> then, the '-' is taken as an operator and this would not be considered an IDL bug.
>
>
>
> C allows:
>
>
>
> short a = -32768;
>
>
>
> but warns about:
>
>
>
> short a = -32769;
>
>
>
> on a machine where a short is 16 bits.

The docs on numeric constants imply that the "+" and "-" are considered operators.

http://www.exelisvis.com/docs/Defining_and_Using_Const.html

This can also be tested experimentally by writing a DLM that checks if an input IDL_VARIABLE is an expression or a constant.
