Subject: Re: New Object Method Invocation Syntax Brokenness
Posted by chris_torrence@NOSPAM on Fri, 10 Jun 2011 20:03:55 GMT
View Forum Message <> Reply to Message

On May 18, 4:02 pm, Chris Torrence <gorth...@gmail.com> wrote:
>
> This is definitely a bug. As Paulo points out, we already fixed the
> problem with the "dot". In IDL 8.0.1 and IDL 8.1 the field always
> takes precedence over the method, unless "compile_opt idl2" is in
> effect. We did this to preserve backwards compatibility - if you had
> code that used parentheses, and there was a field with the same name,
> then IDL will access the field.
>
> However, it looks like the fix went too far and also included the
> arrow. We'll investigate the bug and let you know what we find.
>

Hi all,

We have fixed this bug, and it will be available in the next version of IDL. In summary:

1. If you use the "arrow", IDL will always call the method (even if there is a field with the same name).

2. If you use the "dot", and you have both a method and a field with the same name, IDL will access the field if the compiler cannot tell whether it is a method call or a field.

Examples:
a. self->something(...)  ; always calls the "something" method, regardless of what is inside the parentheses
b. self.something[...]  ; always accesses the field
c. self.something(...) {with compile_opt strictarr}  ; always calls the "something" method, regardless of what is inside the parentheses
d. self.something(keyword=5)  ; calls the method because there is a keyword
e. self.something(0:2)  ; accesses the field because there is an array range
f. self.something(2,4)  ; Ambiguous! IDL will access the field.

Thanks for reporting this issue.

Cheers,

Chris
ITTVIS