
Subject: Re: no backwards compatibility in IDL 5.6
Posted by [JD Smith](#) on Mon, 03 Mar 2003 17:12:24 GMT
[View Forum Message](#) <> [Reply to Message](#)

On Fri, 28 Feb 2003 13:02:15 -0700, JD Smith wrote:

> On Fri, 28 Feb 2003 12:42:42 -0700, David Fanning wrote:
>
>> Pavel Romashkin (pavel_romashkin@hotmail.com) writes:
>>
>>> Why is EXECUTE used in this program? Why can't the value just be
>>> returned from each CASE? Execute will slow it down and as far as I can
>>> tell, does nothing special. There is no code that follows the CASE to
>>> prevent you from returning at any point. Will it not compile in 5.4
>>> with the extra keyword? I thought keyword mismatches are runtime
>>> errors. Am I missing something?
>>
>> Alright, here is why I am using EXECUTE. If I change the code to this:
>>
>> returnValue = 0.0
>> version = Float(!VERSION.Release)
>> IF (version LE 5.5) THEN returnValue = ATAN(imgpart, realpart) \$
>> ELSE returnValue = ATAN(complexNum, /Phase)
>>
>> Then the code won't compile in IDL 5.4, complaining about the PHASE
>> keyword not being defined. :-(
>>
>>
>> P.S. The code *does* compile in IDL 5.5, by the way, even though the
>> PHASE keyword is not defined there, either.
>
> Which is when _STRICT_EXTRA was first introduced. Coincidence?
>
> My conspiracy theory: RSI switched from checking built-in system
> routine keywords at compile to run-time with v5.5. This isn't the
> first time they've done this type of thing: around v5.3, they switched
> from checking the validity of system variables at compile time to run
> time (which was rather convenient, actually).
>
> JD

Just to clarify, for the sake of RSI's hard working engineers: this conspiracy theory, like many others, is entertaining, but entirely untrue. IDL checks at compile time whether *any* keywords are allowed at all. Since ATAN went from having none to having one keyword, this explains the difference.

JD
