Subject: Re: no backwards compatibility in IDL 5.6
Posted by David Fanning on Thu, 27 Feb 2003 13:20:00 GMT
View Forum Message <> Reply to Message

Alexander Rauscher (e9121161@stud4.tuwien.ac.at) writes:

> sorry for posting the same thing twice under different subjects, but i
> think this is important...
>
> many of idl programs have to be adapted due do the non existing
> backwards compatibility of atan (and probably many other functions) .
> one wouldn't expect a change in such a fundamental function. so now
> atan(z, /phase) gives the same result as atan(z) in older versions did,
> where z is (re,im)... this is worse than stupid. this is dangerous.
>
> or does anybody know away to circumvent the new "features" of IDL 5.6?

From the tone (and frequency) of your posts, I presume
you think there is some kind of criminal conspiracy going
on here. Do you have any evidence of it? For example, what
did RSI say when you contacted them about this?

The programmers at RSI are prone to the same knuckleheaded
mistakes as the rest of us, but they don't deliberately
break software (or irritate their customers). Their
livelihood depends on people *buying* software, not
complaining about it. Let us know when you find out
their response to this.

Cheers,

David

--
David W. Fanning, Ph.D.
Fanning Software Consulting, Inc.
Phone: 970-221-0438, E-mail: david@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: no backwards compatibility in IDL 5.6
Posted by James Kuyper on Thu, 27 Feb 2003 15:12:34 GMT
View Forum Message <> Reply to Message

Alexander Rauscher wrote:
>
> sorry for posting the same thing twice under different subjects, but i

> think this is important...
>
> many of idl programs have to be adapted due do the non existing
> backwards compatibility of atan (and probably many other functions) .
> one wouldn't expect a change in such a fundamental function. so now
> atan(z, /phase) gives the same result as atan(z) in older versions did,
> where z is (re,im)... this is worse than stupid. this is dangerous.

As you've described it, that doesn't qualify as an example of backwards incompatibility. If /phase is a new option with 5.6 (and it isn't mentioned in the online help for our 5.4 system), then there won't be any existing code using that option. Therefore, atan(z,/phase) could play Beethoven's 5th, and not be a violation of backwards compatibility.

What would be an example of incompatibility, is if you need to provide the /phase option to get that behavior. Is that what you're saying? If so, I'd agree with you that it is stupid, and dangerous.

Subject: Re: no backwards compatibility in IDL 5.6
Posted by R.G. Stockwell on Thu, 27 Feb 2003 15:13:54 GMT
View Forum Message <> Reply to Message

Alexander Rauscher wrote:
> sorry for posting the same thing twice under different subjects, but i
> think this is important...
>
> many of idl programs have to be adapted due do the non existing
> backwards compatibility of atan (and probably many other functions) .
> one wouldn't expect a change in such a fundamental function. so now
> atan(z, /phase) gives the same result as atan(z) in older versions did,
> where z is (re,im)... this is worse than stupid. this is dangerous.
>
> or does anybody know away to circumvent the new "features" of IDL 5.6?
>
> alex

(Perhaps I am missing something here.)

```
RELEASE         STRING    '5.5a'
IDL> print,atan(complex(4,4))
(      1.44452,     0.123674)
IDL> print,atan(complex(4,4),/phase)
% Keyword PHASE not allowed in call to: ATAN
IDL> print,atan(10,10)
      0.785398

RELEASE         STRING    '5.6'
```

```
IDL> print,atan(complex(4,4))
(      1.44452,     0.123674)
IDL> print,atan(complex(4,4),/phase)
0.785398
IDL> print,atan(10,10)
      0.785398
```

The same thing for the case of atan(complex), and
also for the case of atan([float array]). What has broken?
Also, in 5.6, in the use of atan(10,10, /phase), the "/phase"
is ignored (according to the help file).

It seems to me that functionality has been added in
a way the preserves the previous functionality, so no
legacy code should have any effect.

So I don't think you have to worry.

Cheers,
bob

Subject: Re: no backwards compatibility in IDL 5.6
Posted by Craig Markwardt on Thu, 27 Feb 2003 15:38:07 GMT
View Forum Message <> Reply to Message

David Fanning <david@dfanning.com> writes:

> Alexander Rauscher (e9121161@stud4.tuwien.ac.at) writes:
>
...
>>
>> many of idl programs have to be adapted due do the non existing
>> backwards compatibility of atan (and probably many other functions) .
>> one wouldn't expect a change in such a fundamental function. so now
>> atan(z, /phase) gives the same result as atan(z) in older versions did,
>> where z is (re,im)... this is worse than stupid. this is dangerous.
>>
>> or does anybody know away to circumvent the new "features" of IDL 5.6?
>
> From the tone (and frequency) of your posts, I presume
> you think there is some kind of criminal conspiracy going
> on here. Do you have any evidence of it? For example, what
> did RSI say when you contacted them about this?
>
> The programmers at RSI are prone to the same knuckleheaded
> mistakes as the rest of us, but they don't deliberately
> break software (or irritate their customers). Their

> livelihood depends on people *buying* software, not
> complaining about it. Let us know when you find out
> their response to this.

<rant on>
David, are you saying that RSI *accidentally* changed the functional interface to an established function? No, I think it was quite deliberate. I too was bitten by this change.

The correct way to do this would be to keep the old interface, namely ATAN(z) = complex-arg-of(z), and then add a keyword to ATAN to obtain the mathematically correct arc tangent of a complex number. It's one notch down on the elegance curve, but five notches up on the compatibility curve. And compatibility (= maintainability) is what costs us the big bucks.

The same thing goes for STR_SEP, which is apparently obsolete and will be disappearing in some future release of IDL? What the heck? You're right, why would I want to pay for a package that deliberately rots my working and established code?

I'm sure that RSI argues that the "complex-arg" feature of ATAN was never documented. And yet, it was available in IDL 3.6.1, 4.0.1, 5.0, 5.1, 5.2, 5.3, and 5.4. It's not like this feature was an accident, it was deliberately in the code through 3 major versions of IDL. Like a common-law marriage, this feature is de facto a part of IDL. Why they chose to break compatibility, especially within a minor-version release, is beyond me.

<rant off>

Not-ranting-at-David-ly yours,
Craig

--
---------- ----------
Craig B. Markwardt, Ph.D. EMAIL: craigmnet@cow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
---------- ----------

Subject: Re: no backwards compatibility in IDL 5.6
Posted by Alexander Rauscher on Thu, 27 Feb 2003 15:56:57 GMT
View Forum Message <> Reply to Message

On Thu, 27 Feb 2003, James Kuyper wrote:

*Alexander Rauscher wrote:

*>
*> sorry for posting the same thing twice under different subjects, but i
*> think this is important...
*>
*> many of idl programs have to be adapted due do the non existing
*> backwards compatibility of atan (and probably many other functions) .
*> one wouldn't expect a change in such a fundamental function. so now
*> atan(z, /phase) gives the same result as atan(z) in older versions did,
*> where z is (re,im)... this is worse than stupid. this is dangerous.
*
*As you've described it, that doesn't qualify as an example of backwards
*incompatibility. If /phase is a new option with 5.6 (and it isn't
*mentioned in the online help for our 5.4 system), then there won't be
*any existing code using that option. Therefore, atan(z,/phase) could
*play Beethoven's 5th, and not be a violation of backwards compatibility.
*
*What would be an example of incompatibility, is if you need to provide
*the /phase option to get that behavior. Is that what you're saying? If
*so, I'd agree with you that it is stupid, and dangerous.
*

okay, maybe i didn't explain it correctly:
i do need to provide the keyword to to get the same results as in 5.4
e.g.: i have a complex mr image and i can look at the magnitude image:
tvscl, abs(image)
or at the phase image:
tvscl, atan(image) ; 5.4
tvscl, atan(image, /phase); 5.6

so any code written for 5.4 that uses atan() has to be changed to
atan(,/phase) to work correctly in 5.6.

regards,
alex

Subject: Re: no backwards compatibility in IDL 5.6
Posted by thompson on Thu, 27 Feb 2003 15:58:38 GMT
View Forum Message <> Reply to Message

"R.G. Stockwell" <sorry@noemail.now> writes:

> Alexander Rauscher wrote:
>> sorry for posting the same thing twice under different subjects, but i
>> think this is important...
>>
>> many of idl programs have to be adapted due do the non existing

>> backwards compatibility of atan (and probably many other functions) .
>> one wouldn't expect a change in such a fundamental function. so now
>> atan(z, /phase) gives the same result as atan(z) in older versions did,
>> where z is (re,im)... this is worse than stupid. this is dangerous.
>>
>> or does anybody know away to circumvent the new "features" of IDL 5.6?
>>
>> alex

> (Perhaps I am missing something here.)

> RELEASE STRING '5.5a'
> IDL> print,atan(complex(4,4))
> (1.44452, 0.123674)
> IDL> print,atan(complex(4,4),/phase)
> % Keyword PHASE not allowed in call to: ATAN
> IDL> print,atan(10,10)
> 0.785398

> RELEASE STRING '5.6'
> IDL> print,atan(complex(4,4))
> (1.44452, 0.123674)
> IDL> print,atan(complex(4,4),/phase)
> 0.785398
> IDL> print,atan(10,10)
> 0.785398

> The same thing for the case of atan(complex), and
> also for the case of atan([float array]). What has broken?
> Also, in 5.6, in the use of atan(10,10, /phase), the "/phase"
> is ignored (according to the help file).

> It seems to me that functionality has been added in
> a way the preserves the previous functionality, so no
> legacy code should have any effect.

Evidently, the change in behavior was introduced in 5.5 or 5.5a. This is how it works under 5.4.1 and previous versions of IDL.

IDL> print,atan(complex(4,4))
0.785398

So the point is that you need to change your code to *ADD* the /PHASE keyword to recover the previous behavior. That's definitely a case of something not being backwardly compatible. To make it worse, you have to put in an IF statement to correctly handle the various cases, e.g. 5.4, 5.5, and 5.6, which all act differently.

A better solution may be to separate out the real and imaginary parts, and pass them to ATAN separately, e.g.

```
IDL> x = complex(3,4)
IDL> print,atan(x) ;Under 5.4.1 or below
    0.927295
IDL> print,atan(imaginary(x),float(x))
    0.927295
```

Although you'd still have to worry about the distinction between single and double-precision complex numbers.

Bill Thompson

Subject: Re: no backwards compatibility in IDL 5.6
Posted by David Fanning on Thu, 27 Feb 2003 16:07:57 GMT
View Forum Message <> Reply to Message

Craig Markwardt (craigmnet@cow.physics.wisc.edu) writes:

> David, are you saying that RSI *accidentally* changed the functional
> interface to an established function? No, I think it was quite
> deliberate. I too was bitten by this change.

Whoops! I was afraid that post was going to step on toes. If I hadn't of lost most of my sleep last night with this damn sore throat maybe I wouldn't have been so cranky. :-(

No, what I was saying was I don't know what happened. But if the situation is as you say it is, then I would like to hear RSI's side of the argument along with the (apparently justified) vitriol. The chances of making a deliberate knuckleheaded decision can't be too much less than making an accidental one. I'm just (at the moment) interested in fair play.

Cheers,

David

P.S. And I was also interested in how something like this could be construed as "dangerous", rather than as "terribly, unbelievably annoying". :-)

P.S.S. I know you are not yelling at me. :-)

--

David W. Fanning, Ph.D.
Fanning Software Consulting, Inc.
Phone: 970-221-0438, E-mail: david@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: no backwards compatibility in IDL 5.6
Posted by R.G. Stockwell on Thu, 27 Feb 2003 17:53:07 GMT
View Forum Message <> Reply to Message

William Thompson wrote:
> "R.G. Stockwell" <sorry@noemail.now> writes:
>
>
>> Alexander Rauscher wrote:
>>
>>> sorry for posting the same thing twice under different subjects, but i
>>> think this is important...
>>>
>>> many of idl programs have to be adapted due do the non existing
>>> backwards compatibility of atan (and probably many other functions) .
>>> one wouldn't expect a change in such a fundamental function. so now
>>> atan(z, /phase) gives the same result as atan(z) in older versions did,
>>> where z is (re,im)... this is worse than stupid. this is dangerous.
...
> Evidently, the change in behavior was introduced in 5.5 or 5.5a. This is how
> it works under 5.4.1 and previous versions of IDL.
>
> IDL> print,atan(complex(4,4))
> 0.785398
>
> So the point is that you need to change your code to *ADD* the /PHASE keyword
> to recover the previous behavior. That's definitely a case of something not
> being backwardly compatible. To make it worse, you have to put in an IF
> statement to correctly handle the various cases, e.g. 5.4, 5.5, and 5.6, which
> all act differently.

OUCH!

well, if one has to rewrite a ton a code, i humbly suggest
greply "search-and-replace"-ing all "atan(" with the new user function "user_atan(",
where the "user_atan(" is written correctly.
Through dumb luck I had a tiny function called arg() that i always used, so
this did problem went unnoticed through the version changes. whew.

Cheers,

bob

Subject: Re: no backwards compatibility in IDL 5.6
Posted by Richard Younger on Thu, 27 Feb 2003 18:41:34 GMT
View Forum Message <> Reply to Message

William Thompson wrote:
>
> Evidently, the change in behavior was introduced in 5.5 or 5.5a.
[...]
> A better solution may be to separate out the real and imaginary parts, and
> pass them to ATAN separately, e.g.
>
> IDL> x = complex(3,4)
> IDL> print,atan(x) ;Under 5.4.1 or below
> 0.927295
> IDL> print,atan(imaginary(x),float(x))
> 0.927295
>
> Although you'd still have to worry about the distinction between single and
> double-precision complex numbers.

The behavior was changed in 5.5. It's slower and takes more memory to separate the complex and real parts out. See threads from ~6-8 months ago:

http://groups.google.com/groups?selm=on660zqax4.fsf%40cow.ph ysics.wisc.edu
http://groups.google.com/groups?selm=ony9b7s7vc.fsf%40cow.ph ysics.wisc.edu

5.5 (not coincidentally I think) was also when they introduced the REAL_PART() function, which returns float or double precision depending on the source. Of course, this wasn't in the online documentation, just in the printed update, which is a whole 'nother issue.

Even though the atan() change broke my code, slowed it down, and cost me time, I come down on the "terribly annoying" side rather than calling it "dangerous". I just made sure to test all my critical code before migrating fully to 5.5, and the slowdown only cost my minutes-long runtime a few seconds.

Fortunately, I only have to deal with old versions a little. I can imagine that people maintaining critical code for large department-fulls of licenses would be a little more grumpy about the change.

Best,
Rich Younger

Subject: Re: no backwards compatibility in IDL 5.6
Posted by Mark Hadfield on Thu, 27 Feb 2003 19:00:08 GMT
View Forum Message <> Reply to Message

So, as I understand it, the situation is this:

- For real x & y, ATAN(x) returns the inverse tangent of x and ATAN(x,y) returns the inverse tangent of y/x.

- In versions 5.4 and earlier, ATAN also accepted a complex argument: ATAN(COMPLEX(x,y)) returns the inverse tangent of y/x. Looking at the version 5.4 documentation, one would have to say that this is undocumented, but it was supported over several versions and used by many people.

- In version 5.5, ATAN was overhauled. The IDL 5.5 "What's New" makes interesting reading:

> "In IDL 5.5, new support has been added allowing complex input to ACOS, ASIN, and ATAN. Previously, the inverse transcendental functions ACOS and ASIN did not accept complex input. The ATAN function accepted complex input, Z=X+iY, but incorrectly converted the complex number into the 2-argument ATAN(y, x) form and returned a real result. For ATAN, support has been added for input of two complex arguments....The ATAN function now computes the complex arctangent for complex input. Previously, for a complex number Z=X+iY , internally ATAN(Z) would split Z into its real and imaginary components and compute ATAN(Y, X). IDL code that uses this undocumented behavior should be changed by replacing calls to ATAN(Z) with ATAN(IMAGINARY(Z), REAL_PART(Z))."

- In version 5.6, RSI responded to user feedback by introducing the /PHASE keyword to recover the old, allegedly incorrect behaviour.

So yes, the critics are right: backwards incompatibility has been impaired. But the damage was done in 5.5 and the changes in 5.6 represent an attempt to restore the old behaviour (but you have to add a keyword).

--
Mark Hadfield "Ka puwaha te tai nei, Hoea tatou"
m.hadfield@niwa.co.nz
National Institute for Water and Atmospheric Research (NIWA)

Subject: Re: no backwards compatibility in IDL 5.6
Posted by David Fanning on Thu, 27 Feb 2003 19:17:29 GMT
View Forum Message <> Reply to Message

Mark Hadfield (m.hadfield@niwa.cri.nz) writes:

> So yes, the critics are right: backwards incompatibility has been
> impaired. But the damage was done in 5.5 and the changes in 5.6
> represent an attempt to restore the old behaviour (but you have to add
> a keyword).

I've added an article on my web page with a little
program that produces the same results in IDL 5.4, 5.5,
and 5.6 on my machine.

http://www.dfanning.com/math_tips/atan_change.html

I'd be curious to know if this solves the problem for
everyone:

```
FUNCTION ATAN_COMPLEX_WRAPPER, complexNum

returnValue = 0.0
version = Float(!VERSION.Release)
CASE 1 OF
   (version LE 5.4): ok = Execute('returnValue = ATAN(complexNum)')
   (version EQ 5.5): BEGIN
      realpart = Real_Part(complexNum)
      imgpart = Imaginary(complexNum)
      ok = Execute('returnValue = ATAN(realpart, imgpart)')
      END
   (version GE 5.6): ok = Execute('returnValue = ' + $
       'ATAN(complexNum, /Phase)')
ENDCASE
RETURN, returnValue
END
```

Cheers,

David
--
David W. Fanning, Ph.D.
Fanning Software Consulting, Inc.
Phone: 970-221-0438, E-mail: david@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: no backwards compatibility in IDL 5.6
Posted by notspecified on Thu, 27 Feb 2003 20:03:40 GMT
View Forum Message <> Reply to Message

On Fri, 28 Feb 2003 08:00:08 +1300, "Mark Hadfield"
<m.hadfield@niwa.cri.nz> wrote:

> So, as I understand it, the situation is this:
>
> - For real x & y, ATAN(x) returns the inverse tangent of x and
> ATAN(x,y) returns the inverse tangent of y/x.
>
> - In versions 5.4 and earlier, ATAN also accepted a complex
> argument: ATAN(COMPLEX(x,y)) returns the inverse tangent of
> y/x. Looking at the version 5.4 documentation, one would have to say
> that this is undocumented, but it was supported over several
> versions and used by many people.
>
> - In version 5.5, ATAN was overhauled. The IDL 5.5 "What's New"
> makes interesting reading:
>
>   "In IDL 5.5, new support has been added allowing complex input to
>   ACOS, ASIN, and ATAN. Previously, the inverse transcendental
>   functions ACOS and ASIN did not accept complex input. The ATAN
>   function accepted complex input, Z=X+iY, but incorrectly converted
>   the complex number into the 2-argument ATAN(y, x) form and
>   returned a real result. For ATAN, support has been added for input
>   of two complex arguments....The ATAN function now computes the
>   complex arctangent for complex input. Previously, for a complex
>   number Z=X+iY , internally ATAN(Z) would split Z into its real and
>   imaginary components and compute ATAN(Y, X). IDL code that uses
>   this undocumented behavior should be changed by replacing calls to
>   ATAN(Z) with ATAN(IMAGINARY(Z), REAL_PART(Z))."
>

I think this explains it adequately. In older versions, ATAN with a
complex argument returned a useful number --but the number it
returned didn't happen to be the arctangent of a complex argument!
Perhaps people should take a close look at Abramowitz and Stegun,
equation 4.4.39.

FWIW, if you write a program that uses incorrect, undocumented
behavior, you are asking for trouble. RSI can be blamed for not
providing a fast ARG or PHASE function, but this is a venial sin, at
worst. IMHO.

Matt Feinstein does not include his email address
in the text of usenet postings.

Harvard Law of Automotive Repair: Anything that goes away
by itself will come back by itself.

Subject: Re: no backwards compatibility in IDL 5.6
Posted by Alexander Rauscher on Fri, 28 Feb 2003 06:52:11 GMT
View Forum Message <> Reply to Message

On Thu, 27 Feb 2003, David Fanning wrote:
*
*I've added an article on my web page with a little
*program that produces the same results in IDL 5.4, 5.5,
*and 5.6 on my machine.
*
* http://www.dfanning.com/math_tips/atan_change.html
*
*I'd be curious to know if this solves the problem for
*everyone:
*
* FUNCTION ATAN_COMPLEX_WRAPPER, complexNum
*
* returnValue = 0.0
* version = Float(!VERSION.Release)
* CASE 1 OF
*   (version LE 5.4): ok = Execute('returnValue = ATAN(complexNum)')
*   (version EQ 5.5): BEGIN
*   realpart = Real_Part(complexNum)
*   imgpart = Imaginary(complexNum)
*   ok = Execute('returnValue = ATAN(realpart, imgpart)')
*   END
*   (version GE 5.6): ok = Execute('returnValue = ' + $
*   'ATAN(complexNum, /Phase)')
* ENDCASE
* RETURN, returnValue
* END
*
*Cheers,
*
*David
*--
*David W. Fanning, Ph.D.
*Fanning Software Consulting, Inc.
*Phone: 970-221-0438, E-mail: david@dfanning.com
*Coyote's Guide to IDL Programming: http://www.dfanning.com/
*Toll-Free IDL Book Orders: 1-888-461-0155
*

cool! that looks like a useful solution. especially because one can use a
simple sed script to replace every atan in every *.pro. we could only
test it with 5.4 (yep, we actually "disupgraded". but who knows, with this we
might "undisupgrade" soon...) and it works...

regards,

alex

Subject: Re: no backwards compatibility in IDL 5.6
Posted by Craig Markwardt on Fri, 28 Feb 2003 07:42:13 GMT
View Forum Message <> Reply to Message

notspecified@dev.null (Matt Feinstein) writes:
> I think this explains it adequately. In older versions, ATAN with a
> complex argument returned a useful number --but the number it
> returned didn't happen to be the arctangent of a complex argument!
> Perhaps people should take a close look at Abramowitz and Stegun,
> equation 4.4.39.

Matt, let me say that I totally agree. The original behavior of ATAN was the correct implementation of the incorrect algorithm.

> FWIW, if you write a program that uses incorrect, undocumented
> behavior, you are asking for trouble. RSI can be blamed for not
> providing a fast ARG or PHASE function, but this is a venial sin, at
> worst. IMHO.

Here is where I completely disagree. RSI covered up their original "oops" with another even bigger oops. There is no excuse to break an existing, working, interface in minor-release software. I realize that having ATAN do the correct "Abramowitz & Stegun" thing is more elegant, but I still argue that compatibility and maintainability always trumps elegance, at least in minor releases. RSI had their chance at elegance the first time around.

The original behavior of ATAN with complex numbers was available from IDL 3.6 through IDL 5.4. That's over seven releases, and nearly a decade of stability!

Now let's get to the documentation question. The "old" behavior of ATAN *WAS* in fact documented in the _Using IDL_ manual. A quick check of the Signal Processing chapter, under the section "Magnitude and Phase" shows this example:

```
V = FFT(U)
...
; Phase of first half of v:
phi = ATAN(V(0:N/2))
```

There it is right there! [It's also indexed under "phase, signal spectra".] Five years ago when I was a novice on FFTs and IDL, I went through these examples, and picked up on the techniques described

there. After all, I was learning from the experts.

Now you may ask, in what versions of the manual did this example appear? As near as I can tell, it showed up in IDL 5.0, and kept appearing up through 5.4. And STILL appeared in IDL 5.5. And *STILL* appears today in IDL 5.6!!!

This posting is not really about ATAN. It's really about how RSI appears to be making unilateral decisions about the IDL language which breaks the compatibility of peoples' code. They should stop that.

Craig

--

---------- ----------
Craig B. Markwardt, Ph.D. EMAIL: craigmnet@cow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
---------- ----------

Subject: Re: no backwards compatibility in IDL 5.6
Posted by R.G. Stockwell on Fri, 28 Feb 2003 13:59:05 GMT
View Forum Message <> Reply to Message

Craig Markwardt wrote:
> notspecified@dev.null (Matt Feinstein) writes:
>
>> I think this explains it adequately. In older versions, ATAN with a
>> complex argument returned a useful number --but the number it
>> returned didn't happen to be the arctangent of a complex argument!
>> Perhaps people should take a close look at Abramowitz and Stegun,
>> equation 4.4.39.
>
>
> Matt, let me say that I totally agree. The original behavior of ATAN
> was the correct implementation of the incorrect algorithm.
>
>
>> FWIW, if you write a program that uses incorrect, undocumented
>> behavior, you are asking for trouble. RSI can be blamed for not
>> providing a fast ARG or PHASE function, but this is a venial sin, at
>> worst. IMHO.
>
>
> Here is where I completely disagree. RSI covered up their original
> "oops" with another even bigger oops. There is no excuse to break an
> existing, working, interface in minor-release software. I realize
> that having ATAN do the correct "Abramowitz & Stegun" thing is more

> elegant, but I still argue that compatibility and maintainability
> always trumps elegance, at least in minor releases. RSI had their
> chance at elegance the first time around.
...
>
> Craig
>

Be it foolish, but I don't agree with you. I think RSI did the right
thing by "finally" correcting the behaviour of the atan(), even if it
does break existing code. Because now all the user's bitch and moan :),
and fix their code, and never have a problem again with atan().
If they left it as it was, then for the next 100 years, new users would
be bitching and moaning about how the atan() doesn't work for
complex numbers.
As a case in point, check out recent discussions on the /center keyword
in convol. :)

Cheers,
bob

PS, of course, fixing it correctly the first time would have been the thing
to do, rather than requiring a case statement to handle the different IDL versions. :)

And there is always the case to be made that an ATAN function should only
calculate the atan, and the argument of a complex number should be a seperate
modular function.

Subject: Re: no backwards compatibility in IDL 5.6
Posted by notspecified on Fri, 28 Feb 2003 15:04:55 GMT
View Forum Message <> Reply to Message

On Fri, 28 Feb 2003 06:59:05 -0700, "R.G. Stockwell"
<sorry@noemail.now> wrote:

> Craig Markwardt wrote:
>> notspecified@dev.null (Matt Feinstein) writes:
>>
>>> I think this explains it adequately. In older versions, ATAN with a
>>> complex argument returned a useful number --but the number it
>>> returned didn't happen to be the arctangent of a complex argument!
>>> Perhaps people should take a close look at Abramowitz and Stegun,
>>> equation 4.4.39.
>>
>>
>> Matt, let me say that I totally agree. The original behavior of ATAN

>> was the correct implementation of the incorrect algorithm.
>>
>>
>>> FWIW, if you write a program that uses incorrect, undocumented
>>> behavior, you are asking for trouble. RSI can be blamed for not
>>> providing a fast ARG or PHASE function, but this is a venial sin, at
>>> worst. IMHO.
>>
>>
>> Here is where I completely disagree. RSI covered up their original
>> "oops" with another even bigger oops. There is no excuse to break an
>> existing, working, interface in minor-release software. I realize
>> that having ATAN do the correct "Abramowitz & Stegun" thing is more
>> elegant, but I still argue that compatibility and maintainability
>> always trumps elegance, at least in minor releases. RSI had their
>> chance at elegance the first time around.
> ...
>>
>> Craig
>>
>
>
> Be it foolish, but I don't agree with you. I think RSI did the right
> thing by "finally" correcting the behaviour of the atan(), even if it
> does break existing code. Because now all the user's bitch and moan :),
> and fix their code, and never have a problem again with atan().
> If they left it as it was, then for the next 100 years, new users would
> be bitching and moaning about how the atan() doesn't work for
> complex numbers.
> As a case in point, check out recent discussions on the /center keyword
> in convol. :)

There is a classic essay entitled 'The Rise of Worse is Better' by
Richard Gabriel on how different software design philosophies balance
requirements for simplicity, correctness, consistency, and
completeness. Gabriel isn't shy about arguing for his own point of
view ('Incorrectness is not allowed'), and caricatures the views of
people who disagree-- but the questions raised are fundamental, and
people -will- disagree about these things.

My own view is that incorrectness -has- to be fixed. I understand
that, since we all make mistakes at random, this means inevitable,
random, and sometimes painful bumps in the road for simplicity and
consistency. But, to me, the alternatives seem worse.

Matt Feinstein   does not include his email address
                in the text of usenet postings.

Harvard Law of Automotive Repair: Anything that goes away
by itself will come back by itself.

Subject: Re: no backwards compatibility in IDL 5.6
Posted by Pavel Romashkin on Fri, 28 Feb 2003 17:25:43 GMT
View Forum Message <> Reply to Message

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?
Pavel

David Fanning wrote:
>
> FUNCTION ATAN_COMPLEX_WRAPPER, complexNum
>
> returnValue = 0.0
> version = Float(!VERSION.Release)
> CASE 1 OF
> (version LE 5.4): ok = Execute('returnValue = ATAN(complexNum)')
> (version EQ 5.5): BEGIN
> realpart = Real_Part(complexNum)
> imgpart = Imaginary(complexNum)
> ok = Execute('returnValue = ATAN(realpart, imgpart)')
> END
> (version GE 5.6): ok = Execute('returnValue = ' + $
> 'ATAN(complexNum, /Phase)')
> ENDCASE
> RETURN, returnValue
> END

Subject: Re: no backwards compatibility in IDL 5.6
Posted by David Fanning on Fri, 28 Feb 2003 17:47:52 GMT
View Forum Message <> Reply to Message

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?

I don't know. I got so confused with the discussion
yesterday I finally just said the hell with it and
went back to bed. :-(

Let's just say I had no idea so many people used
the ATAN function.

I'm totally confused about when things will compile
and when they won't. The only thing I know for sure
is they won't compile if they have to. For example, they
would never compile if you were doing a demo in front
of the new Vice President of the company.

I think there must have been a change somewhere along
the way (while we are on this subject). Because I didn't
expect that file to compile in IDL 5.4, due to the REAL_PART
function in the IDL 5.5 part of the CASE statement. When it
did, that's when I realized I needed a nap.

Anyway, why don't you fix it up, Pavel, and I'll post
the darn thing. :-)

Cheers,

David

--
David W. Fanning, Ph.D.
Fanning Software Consulting, Inc.
Phone: 970-221-0438, E-mail: david@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: no backwards compatibility in IDL 5.6
Posted by JD Smith on Fri, 28 Feb 2003 17:52:48 GMT
View Forum Message <> Reply to Message

On Fri, 28 Feb 2003 10:47:52 -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?
>
> I don't know. I got so confused with the discussion yesterday I finally
> just said the hell with it and went back to bed. :-(
>
> Let's just say I had no idea so many people used the ATAN function.
>
> I'm totally confused about when things will compile and when they won't.
> The only thing I know for sure is they won't compile if they have to.
> For example, they would never compile if you were doing a demo in front
> of the new Vice President of the company.
>
> I think there must have been a change somewhere along the way (while we
> are on this subject). Because I didn't expect that file to compile in
> IDL 5.4, due to the REAL_PART function in the IDL 5.5 part of the CASE
> statement. When it did, that's when I realized I needed a nap.
>
>
I bet you didn't change your IDL PATH between running 5.5 and 5.4.
REAL_PART is in the !DIR/lib as a .pro file, and 5.4 can use it just as
well. Also, resolving a routine call into a compiled .pro file occurs at
run-time, so even if you called it REAL_PART_DOESNT_EXIST it would still
compile.

I.e.:

```
pro foo
  if 0 then MY_NON_EXISTENT_PROCEDURE,4
end
```

would compile and run perfectly fine.

IDL does check the number of arguments of *built-in* (i.e. not .pro)
system routines at compile time (this counts DLMs too). I think Pavel is
right that all keywords are checked at run-time.

JD

Subject: Re: no backwards compatibility in IDL 5.6
Posted by Pavel Romashkin on Fri, 28 Feb 2003 17:55:23 GMT
View Forum Message <> Reply to Message

David Fanning wrote:
>

> I'm totally confused about when things will compile
> and when they won't. The only thing I know for sure
> is they won't compile if they have to. For example, they
> would never compile if you were doing a demo in front
> of the new Vice President of the company.

I know what you are saying.
In this case I think it is best to make the demo in the form of a
PowerPoint presentation. Besides, it gives you a great chance to use
Photoshop to make sure the images you are presenting look the way they
are supposed to :-)
Cheers,
Pavel

P.S. I have not used ATAN ever in my code. My programs are just way too simple-minded.

Subject: Re: no backwards compatibility in IDL 5.6
Posted by David Fanning on Fri, 28 Feb 2003 19:42:42 GMT
View Forum Message <> Reply to Message

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. :-(

Unless I've completely misunderstood (probable with the
drugs I'm taking for this cold), keywords should be
sorted out at run-time, not at compile time, according
to your theory. What then to you make of this?

Cheers,

David

P.S. The code *does* compile in IDL 5.5, by the way,
even though the PHASE keyword is not defined there, either.

--
David W. Fanning, Ph.D.
Fanning Software Consulting, Inc.
Phone: 970-221-0438, E-mail: david@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: no backwards compatibility in IDL 5.6
Posted by thompson on Fri, 28 Feb 2003 20:12:17 GMT
View Forum Message <> Reply to Message

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?
> Pavel

Yes, without the execute statement, it will not compile in versions earlier
than 5.4. You get the error message

```
IDL> .run atan_complex_wrapper

      (version GE 5.6): returnValue = ATAN(complexNum, /Phase)
                                                 ^
% Keyword parameters not allowed in call.
  At: /disk1/thompson/atan_complex_wrapper.pro, Line 12
% 1 Compilation errors in module ATAN_COMPLEX_WRAPPER.
```

However, only the last execute statement is actually required. The first two,
without the new keyword, can be direct statements.

Bill Thompson

> David Fanning wrote:
>>
>> FUNCTION ATAN_COMPLEX_WRAPPER, complexNum

```
>>
>>    returnValue = 0.0
>>    version = Float(!VERSION.Release)
>>    CASE 1 OF
>>       (version LE 5.4): ok = Execute('returnValue = ATAN(complexNum)')
>>       (version EQ 5.5): BEGIN
>>          realpart = Real_Part(complexNum)
>>          imgpart = Imaginary(complexNum)
>>          ok = Execute('returnValue = ATAN(realpart, imgpart)')
>>          END
>>       (version GE 5.6): ok = Execute('returnValue = ' + $
>>           'ATAN(complexNum, /Phase)')
>>    ENDCASE
>>    RETURN, returnValue
>>    END
```

Subject: Re: no backwards compatibility in IDL 5.6
Posted by thompson on Fri, 28 Feb 2003 20:15:48 GMT
View Forum Message <> Reply to Message

David Fanning <david@dfanning.com> writes:

(stuff deleted)

> I think there must have been a change somewhere along
> the way (while we are on this subject). Because I didn't
> expect that file to compile in IDL 5.4, due to the REAL_PART
> function in the IDL 5.5 part of the CASE statement. When it
> did, that's when I realized I needed a nap.

Sure it compiled. Even REAL_PART isn't a built-in function in IDL 5.4, the compiler just assumes there's going to be a real_part.pro procedure somewhere, but it won't look for that until runtime.

Hope you had a good nap, ;^)

Bill Thompson
