Subject: Re: Resolving Built-ins and FORWARD_FUNCTION
Posted by Craig Markwardt on Tue, 09 Dec 2003 06:15:20 GMT
View Forum Message <> Reply to Message

JD Smith <jdsmith@as.arizona.edu> writes:

> Sent to RSI:
>
> ==
===============
> Using FORWARD_FUNCTION creates an unresolved stub in the routine list,
> even for built-in routines. E.g., in the NasaLib WRITEFITS you find :
...
> IDL could either check for built-in's being used in FORWARD_FUNCTION, or
> RESOLVE_ROUTINE could do the same, or FORWARD_FUNCTION functions could
> be removed from the list once they are encountered in the file. Since
> you can't override a built-in command (like FILE_SEARCH) with any amount
> of !PATH fiddling, it makes sense not to put built-ins on the unresolved
> list via FORWARD_FUNCTION.
> ==
===============

Yes, these seems like a totally legit complaint. I don't think this
problem showed up before IDL 6.0, did it? Or else, why did nobody
complain before now?

> Also, does anyone know what a SAV file run in the IDLVM does with a
> statement like:
>
> source=routine_info('MyPro',/SOURCE)
>
> I use these types of constructs to locate data bundled with my source
> distribution, and I want it to work with the IDLVM too. Since the VM
> technically doesn't do any compiling of files, I presume it might not
> do any path searching for file source either, in which case I'd have
> to come up with something different.

Why not try it yourself? I found that the "source" it reported was
the path of the .sav file.

Craig

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

-- --------------

Subject: Re: Resolving Built-ins and FORWARD_FUNCTION
Posted by JD Smith on Tue, 09 Dec 2003 19:29:26 GMT
View Forum Message <> Reply to Message

On Mon, 08 Dec 2003 23:15:20 -0700, Craig Markwardt wrote:

> JD Smith <jdsmith@as.arizona.edu> writes:
>
>
>> Sent to RSI:
>>
>> ==
===============
>> Using FORWARD_FUNCTION creates an unresolved stub in the routine list,
>> even for built-in routines. E.g., in the NasaLib WRITEFITS you find :
> ...
>> IDL could either check for built-in's being used in FORWARD_FUNCTION,
>> or RESOLVE_ROUTINE could do the same, or FORWARD_FUNCTION functions
>> could be removed from the list once they are encountered in the file.
>> Since you can't override a built-in command (like FILE_SEARCH) with any
>> amount of !PATH fiddling, it makes sense not to put built-ins on the
>> unresolved list via FORWARD_FUNCTION.
>> ==
===============
>
> Yes, these seems like a totally legit complaint. I don't think this
> problem showed up before IDL 6.0, did it? Or else, why did nobody
> complain before now?

Thanks Craig. RSI has filed an internal bug-fix request on this one,
and suggested a workaround of using "COMPILE_OPT IDL2" in place of
FORWARD_FUNCTION. Of course, this would not really help Wayne, who is
using FORWARD_FUNCTION to allow NasaLib to run for older IDL 5.x
versions (COMPILE_OPT was introduced in v5.3). And it also doesn't
help if you're "compiling" in code from libraries over which you have
no control.

I'm not sure why nobody complained: the bug is present as far back as
v5.5 (which is the earliest version I had to test). The test is easy,
if you have AstroLib:

```
IDL> .run writefits
IDL> resolve_all
```

will give an error.

>> Also, does anyone know what a SAV file run in the IDLVM does with a
>> statement like:
>>
>> source=routine_info('MyPro',/SOURCE)
>>
>> I use these types of constructs to locate data bundled with my source
>> distribution, and I want it to work with the IDLVM too. Since the VM
>> technically doesn't do any compiling of files, I presume it might not
>> do any path searching for file source either, in which case I'd have to
>> come up with something different.
>
> Why not try it yourself? I found that the "source" it reported was the
> path of the .sav file.

Because I knew I could get you to do it for me ;). Next question I
should probably find out for myself: is there a programmatic way to
tell if you're running from a restored SAV file or from real, live
source? Can you tell I've almost never built a routine SAV file? I'm
trying to see if I can get a large package to run with the IDLVM.

Thanks,

JD

Subject: Re: Resolving Built-ins and FORWARD_FUNCTION
Posted by Craig Markwardt on Tue, 09 Dec 2003 19:47:20 GMT
View Forum Message <> Reply to Message

JD Smith <jdsmith@as.arizona.edu> writes:
> Thanks Craig. RSI has filed an internal bug-fix request on this one,
> and suggested a workaround of using "COMPILE_OPT IDL2" in place of
> FORWARD_FUNCTION. Of course, this would not really help Wayne, who is
> using FORWARD_FUNCTION to allow NasaLib to run for older IDL 5.x
> versions (COMPILE_OPT was introduced in v5.3). And it also doesn't
> help if you're "compiling" in code from libraries over which you have
> no control.
>
> I'm not sure why nobody complained: the bug is present as far back as
> v5.5 (which is the earliest version I had to test). The test is easy,
> if you have AstroLib:

I confirmed the problem is present on IDL's 5.1 through 5.4 as well.

> Because I knew I could get you to do it for me ;). Next question I
> should probably find out for myself: is there a programmatic way to
> tell if you're running from a restored SAV file or from real, live
> source? Can you tell I've almost never built a routine SAV file? I'm
> trying to see if I can get a large package to run with the IDLVM.

You could probably look for .sav/.SAV in the source file name?

I really don't use IDL save files for routines any more, after a
certain large corporation (name similar to Kodiak) threatened a
potential lawsuit against me.

Craig

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

Subject: Re: Resolving Built-ins and FORWARD_FUNCTION
Posted by JD Smith on Tue, 09 Dec 2003 20:57:50 GMT
View Forum Message <> Reply to Message

On Tue, 09 Dec 2003 12:47:20 -0700, Craig Markwardt wrote:

> JD Smith <jdsmith@as.arizona.edu> writes:
>> Thanks Craig. RSI has filed an internal bug-fix request on this one,
>> and suggested a workaround of using "COMPILE_OPT IDL2" in place of
>> FORWARD_FUNCTION. Of course, this would not really help Wayne, who is
>> using FORWARD_FUNCTION to allow NasaLib to run for older IDL 5.x
>> versions (COMPILE_OPT was introduced in v5.3). And it also doesn't
>> help if you're "compiling" in code from libraries over which you have
>> no control.
>>
>> I'm not sure why nobody complained: the bug is present as far back as
>> v5.5 (which is the earliest version I had to test). The test is easy,
>> if you have AstroLib:
>
> I confirmed the problem is present on IDL's 5.1 through 5.4 as well.
>
>
>
>> Because I knew I could get you to do it for me ;). Next question I
>> should probably find out for myself: is there a programmatic way to
>> tell if you're running from a restored SAV file or from real, live

>> source? Can you tell I've almost never built a routine SAV file? I'm
>> trying to see if I can get a large package to run with the IDLVM.
>
> You could probably look for .sav/.SAV in the source file name?
>
> I really don't use IDL save files for routines any more, after a certain
> large corporation (name similar to Kodiak) threatened a potential
> lawsuit against me.

Yes, I think we all were aghast that you'd be targeted in that way.
Also amazing was the notion that the simple .sav format could be
considered a copyright protection device. But if you want your users
to have access through the IDLVM, there is no other way.

Thanks,

JD

Subject: Re: Resolving Built-ins and FORWARD_FUNCTION
Posted by Wayne Landsman on Tue, 09 Dec 2003 21:05:23 GMT
View Forum Message <> Reply to Message

> I'm not sure why nobody complained: the bug is present as far back as
> v5.5 (which is the earliest version I had to test). The test is easy,
> if you have AstroLib:
>
> IDL> .run writefits
> IDL> resolve_all
>
> will give an error.
>

I'm sure this has been mentioned but another workaround is to use

IDL> resolve_all,/continue_on_error

which is what I've been using by default. The disadvantage is that
you might not recognize when a real procedure is missing. --Wayne

---

Subject: Re: Resolving Built-ins and FORWARD_FUNCTION
Posted by JD Smith on Tue, 09 Dec 2003 23:46:27 GMT
View Forum Message <> Reply to Message

On Tue, 09 Dec 2003 14:05:23 -0700, Wayne Landsman wrote:

>> I'm not sure why nobody complained: the bug is present as far back as
>> v5.5 (which is the earliest version I had to test). The test is easy,
>> if you have AstroLib:
>>
>> IDL> .run writefits
>> IDL> resolve_all
>>
>> will give an error.
>>
>>
> I'm sure this has been mentioned but another workaround is to use
>
> IDL> resolve_all,/continue_on_error
>
> which is what I've been using by default.   The disadvantage is that
> you might not recognize when a real procedure is missing.   --Wayne

Thanks Wayne. I was wondering why FORWARD_FUNCTION was needed at all,
so I took a look. I think I've found the reason: when IDL compiles a
routine, it takes any function call with keyword arguments (which it
knows is a function call, and not a variable using the old ()
indexing syntax), and checks that it really exists somewhere as a
function... it doesn't compile the function, it just checks that it
exists. Example:

```
;; Compiles fine, since it could be an indexed variable, for all IDL
;; knows
pro testff
  if !PI eq 2.0 then ret=unknown_function(b)
end

;; Syntax error on compile, since it's not a known function, and
;; indexing statements shouldn't have KEYWORDS in them!
pro testff
  if !PI eq 2.0 then ret=unknown_function(b,TEST=2)
end

;; Compiles fine
pro testff
  FORWARD_FUNCTION unknown_function
  if !PI eq 2.0 then ret=unknown_function(b,TEST=2)
end

;; Compiles fine
pro testff
  COMPILE_OPT IDL2
  if !PI eq 2.0 then ret=unknown_function(b,TEST=2)
```

end

Notice that it doesn't matter that the function will *never* be called
(in Euclidean universes, anyway). Also, amusingly, IDL really only
checks that the function in question is built-in or that there is a
file named "unknown_function.pro" somewhere on the !PATH: this could
contain a procedure named "unknown_function", or your grandmother's
bourbon fruitcake recipe, so long as it exists.

If you call a function with keyword arguments which doesn't exists
(e.g. because it's available only in a later version of IDL), you can
use FORWARD_FUNCTION to keep IDL from issuing a syntax error,
otherwise it will insist that, if it's not a function, it must be an
indexing statement, with a keyword-like syntax error inside.

Interestingly, if you use "COMPILE_OPT IDL2", this type of error
disappears. This is because IDL no longer needs to go over every
thing that looks like foo(), and check that it's either a function or
a syntax-error-free indexing statment. It just assumes it's a
function. This probably speeds up compiling just a bit too, so I'd go
as far as to say COMPILE_OPT IDL2 is probably a better all-around
solution than FORWARD_FUNCTION, unless you require IDL<v5.3
compatibility.

JD

Subject: Re: Resolving Built-ins and FORWARD_FUNCTION
Posted by JD Smith on Wed, 10 Dec 2003 23:58:06 GMT
View Forum Message <> Reply to Message

On Tue, 09 Dec 2003 12:47:20 -0700, Craig Markwardt wrote:

> JD Smith <jdsmith@as.arizona.edu> writes:
>> should probably find out for myself: is there a programmatic way to
>> tell if you're running from a restored SAV file or from real, live
>> source? Can you tell I've almost never built a routine SAV file? I'm
>> trying to see if I can get a large package to run with the IDLVM.
>
> You could probably look for .sav/.SAV in the source file name?

Found this:

http://www.rsinc.com/services/techtip.asp?ttid=3564

i.e. LMGR(/VM).

JD

Subject: Re: Resolving Built-ins and FORWARD_FUNCTION
Posted by David Fanning on Thu, 11 Dec 2003 00:27:21 GMT
View Forum Message <> Reply to Message

JD Smith writes:

> Found this:
>
> http://www.rsinc.com/services/techtip.asp?ttid=3564
>
> i.e. LMGR(/VM).

Well, as long as you are doing experiments again, does
LMGR(/VM) return the same thing as LMGR(/RUNTIME)? I
thought I needed to know this the other day for some
reason, and I was too lazy to figure it out. :-)

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: Resolving Built-ins and FORWARD_FUNCTION
Posted by JD Smith on Thu, 11 Dec 2003 01:26:23 GMT
View Forum Message <> Reply to Message

On Wed, 10 Dec 2003 17:27:21 -0700, David Fanning wrote:

> JD Smith writes:
>
>> Found this:
>>
>> http://www.rsinc.com/services/techtip.asp?ttid=3564
>>
>> i.e. LMGR(/VM).
>
> Well, as long as you are doing experiments again, does LMGR(/VM) return

> the same thing as LMGR(/RUNTIME)? I thought I needed to know this the
> other day for some reason, and I was too lazy to figure it out. :-)

I don't know, but apparently you can use any keyword combo in a Boolean
OR sense, e.g LMGR(/VM,/RUNTIME) to find out if either is true.

JD

Subject: Re: Resolving Built-ins and FORWARD_FUNCTION
Posted by David Fanning on Thu, 11 Dec 2003 03:17:58 GMT

View Forum Message <> Reply to Message

JD Smith writes:

> I don't know, but apparently you can use any keyword combo in a Boolean
> OR sense, e.g LMGR(/VM,/RUNTIME) to find out if either is true.

See, JD, I'm pretty sure you are the only one of the
regulars here who would have figured this out. It pays to
let you do the work. ;-)

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: Resolving Built-ins and FORWARD_FUNCTION
Posted by David Fanning on Thu, 11 Dec 2003 03:49:11 GMT

View Forum Message <> Reply to Message

JD Smith writes:

>> Well, as long as you are doing experiments again, does LMGR(/VM) return
>> the same thing as LMGR(/RUNTIME)? I thought I needed to know this the
>> other day for some reason, and I was too lazy to figure it out. :-)
>
> I don't know, but apparently you can use any keyword combo in a Boolean
> OR sense, e.g LMGR(/VM,/RUNTIME) to find out if either is true.

While this is true, it didn't answer my original question, so

I just thought I'd look into it myself. :-)

In a save file running on a RUNTIME license:

```
   LMGR(/Runtime) is set to 1
   LMGR(/VM) is set to 0
```

In the same save file running on the VM:

```
   LMGR(/Runtime) is set to 1
   LMGR(/VM) is set to 1
```

Go figure. :-(

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