Subject: Re: Technique for "method\_missing" in IDL objects Posted by David Fanning on Fri, 17 Jul 2009 16:22:17 GMT

View Forum Message <> Reply to Message

#### Paul van Delst writes:

- > I'm stringing together a bunch of different objects into a container. When I define
- > objects I always define an "Inspect" method so I can have a lookee at the internals of the
- > objects (like the ruby inspect method). However, the inspect method for the container
- > simply loops over the objects that have been placed in it calling their inspect methods.
- > So, if I reach an object that does not have an inspect method, is there a technique to
- > pre-determine if I can even call the method to avoid the

>

> % Attempt to call undefined procedure/function: 'OBJ::INSPECT'

>

> error that I get?

>

- > Ruby provides a "method\_missing" method to enable one to handle this sort of thing. Does
- > IDL have any sort of equivalent? Looking at the various ROUTINE\_INFO,

# RESOLVE ROUTINE

> help, they don't appear to be that reliable.

## OBJ\_HASMETHOD might work:

http://www.dfanning.com/code\_tips/hasmethod.html

Cheers,

David

--

David Fanning, Ph.D.

Fanning Software Consulting, Inc.

Coyote's Guide to IDL Programming: http://www.dfanning.com/

Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: Technique for "method\_missing" in IDL objects Posted by mankoff on Fri, 17 Jul 2009 16:55:32 GMT

View Forum Message <> Reply to Message

On Jul 17, 12:20 pm, Paul van Delst <paul.vande...@noaa.gov> wrote:

- > Hello,
- >
- > I'm stringing together a bunch of different objects into a container. When I define
- > objects I always define an "Inspect" method so I can have a lookee at the internals of the
- > objects (like the ruby inspect method). However, the inspect method for the container
- > simply loops over the objects that have been placed in it calling their inspect methods.

- So, if I reach an object that does not have an inspect method, is there a technique to
  pre-determine if I can even call the method to avoid the
  % Attempt to call undefined procedure/function: 'OBJ::INSPECT'
  error that I get?
  Ruby provides a "method\_missing" method to enable one to handle this sort of thing. Does
  IDL have any sort of equivalent? Looking at the various ROUTINE\_INFO,
  RESOLVE\_ROUTINE
  help, they don't appear to be that reliable.
  cheers,
- > cheers,
  > paulv

If a top level object has an INSPECT method, then it will always be found in a parent. I'm not sure if IDL objects have this ability, but can the top level INSPECT check to see if it was called by itself or a child? If so, and you never call it explicitly, then if a child is running the top level INSPECT it implies the child does not have its own INSPECT.

-k.

>

>>

>>

>>

Subject: Re: Technique for "method\_missing" in IDL objects Posted by Paul Van Delst[1] on Fri, 17 Jul 2009 18:07:34 GMT View Forum Message <> Reply to Message

## David Fanning wrote:

- > Paul van Delst writes:
- >> I'm stringing together a bunch of different objects into a container. When I define
- >> objects I always define an "Inspect" method so I can have a lookee at the internals of the
- >> objects (like the ruby inspect method). However, the inspect method for the container
- >> simply loops over the objects that have been placed in it calling their inspect methods.
- >> So, if I reach an object that does not have an inspect method, is there a technique to
- >> pre-determine if I can even call the method to avoid the
- >> % Attempt to call undefined procedure/function: 'OBJ::INSPECT'
- >> error that I get?
- >> Ruby provides a "method\_missing" method to enable one to handle this sort of thing. Does >> IDL have any sort of equivalent? Looking at the various ROUTINE\_INFO,

# RESOLVE\_ROUTINE

>> help, they don't appear to be that reliable.

```
> OBJ_HASMETHOD might work:
> http://www.dfanning.com/code_tips/hasmethod.html
Ah! Perfect. That's exactly what I was looking for!
cheers,
pauly
```

Subject: Re: Technique for "method\_missing" in IDL objects Posted by Paul Van Delst[1] on Fri, 17 Jul 2009 18:15:14 GMT View Forum Message <> Reply to Message

```
mankoff wrote:
```

```
> On Jul 17, 12:20 pm, Paul van Delst <paul.vande...@noaa.gov> wrote: >> Hello,
```

>>

- >> I'm stringing together a bunch of different objects into a container. When I define
- >> objects I always define an "Inspect" method so I can have a lookee at the internals of the
- >> objects (like the ruby inspect method). However, the inspect method for the container
- >> simply loops over the objects that have been placed in it calling their inspect methods.
- >> So, if I reach an object that does not have an inspect method, is there a technique to
- >> pre-determine if I can even call the method to avoid the

>> >>

>>

% Attempt to call undefined procedure/function: 'OBJ::INSPECT'

>> error that I get?

>>

- >> Ruby provides a "method\_missing" method to enable one to handle this sort of thing. Does
- >> IDL have any sort of equivalent? Looking at the various ROUTINE\_INFO,

#### RESOLVE\_ROUTINE

>> help, they don't appear to be that reliable.

>>

>> cheers,

>>

>> paulv

>

- > If a top level object has an INSPECT method, then it will always be
- > found in a parent. I'm not sure if IDL objects have this ability, but
- > can the top level INSPECT check to see if it was called by itself or a
- > child? If so, and you never call it explicitly, then if a child is
- > running the top level INSPECT it implies the child does not have its
- > own INSPECT.

Umm... not sure I follow. FWIW, here's my container Inspect method:

```
PRO LBL_Input::Inspect, $
 Verbose=Verbose, $; Input keyword
 Debug=Debug
                   ; Input keyword
 IF ( KEYWORD_SET(Debug) ) THEN HELP, /ROUTINES
 HELP, self, /OBJECTS
 IF (KEYWORD_SET(Verbose)) THEN BEGIN
  obj = self->Get(/ALL, COUNT=n_Objs)
  IF (n Objs EQ 0) THEN RETURN
  ; Loop over contained objects
  FOR n = 0L, n_Objs-1L DO BEGIN
   obi[n]->Inspect, Verbose=Verbose, Debug=Debug
  ENDFOR
 ENDIF
END; PRO LBL_Input::Inspect
And it's the call to the individual object inspect methods in the loop,
 obj[n]->Inspect, Verbose=Verbose, Debug=Debug
that can cause the constipation. I changed the above line to
 IF ( OBJ HASMETHOD(obj[n], 'INSPECT') ) THEN $
  obi[n]->Inspect, Verbose=Verbose, Debug=Debug
and now all is wonderful! Woohoo!
:0)
cheers,
paulv
```