Subject: Object Programming
Posted by David Fanning on Wed, 11 May 2011 19:04:19 GMT
View Forum Message <> Reply to Message

Folks,

I guess it has been awhile since I did some extensive object programming, but goodness, IDL doesn't like it when you make mistakes!

I suppose I crashed IDL 25-30 times already today, both IDL 7.1.1 and IDL 8.1. It seems as though if put a breakpoint in an object program and then do a .RESET from the IDL command line that IDL is certain to crash.

The error I was tracking down looked something like this:

```
FUNCTION foo, x, y

   obj = Obj_New('foo', x, y)

   RETURN, obj

END
```

Both routines are written in such a way that you can pass one or two parameters. If I pass a single parameter:

```
IDL> obj = FOO(data)
```

The object INIT method was reporting that I was passing *two* parameters (I was checking with N_Params()) and my program was getting into trouble because I assumed y was defined in the code. It was actually passing an undefined variable into the object INIT method. Is this how IDL has always worked? I could have sworn I've done this many, many times in the past without difficulty, but perhaps I wasn't using N_Params() to check parameters in that code.

Anyway, lesson for today: Don't make a mistake when you are writing IDL object programs!

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: Object Programming
Posted by David Fanning on Thu, 12 May 2011 19:45:01 GMT
View Forum Message <> Reply to Message

Paulo Penteado writes:

>
> On May 11, 4:23 pm, David Fanning <n...@idlcoyote.com> wrote:
>> I guess that's right. I usually check to see if parameters
>> are undefined or not. In this case, because the actual programs
>> are wrappers to the PLOT command, I wanted to know how many
>> there were. Odd that I haven't run into this before, though. :-)
>
> Do you DG's plot (what I would guess by 'command')?

Yes, these are direct graphics objects. Fast, simple, etc.

> I ask because I know there are some peculiar pitfalls in NG's plot():
> instead of (what I expected) every class inheriting Graphic, and being
> able to inherit from them, in the usual way, it is a very convoluted
> system. I suspect the cause was to make it work while minimizing the
> changes or additions to the way the iTools were organized.
>
> The most confusing part is how the classes (like Plot) are created:
> The functions (like plot()) are not the usual init functions, they are
> separate functions that create a Graphic object (itself not created by
> a init function), informing it of what kind of graphic to make. The
> graphic() function creates the proper iTool, and and an object of the
> proper class, which is just a wrapper and contains the Graphic object
> inside.
>
> This is why I gave up on trying to inherit Window for a class, and
> contained it instead (you can see it at
> http://www.ppenteado.net/idl/pp_lib/doc/pp_multiplot__define .html). I
> do not remember whether it would be impossible or just complicated to
> inherit without changing the code in some of IDL's routines (which
> would be a very weird inheritance, if I had to change other code).

Say what!? The fact that I don't even understand *questions* about
function graphics, to say nothing of function graphics themselves,
is one of the reasons I am writing these CGS graphics objects. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: Object Programming
Posted by penteado on Thu, 12 May 2011 19:45:30 GMT
View Forum Message <> Reply to Message

On May 12, 4:41 pm, Paulo Penteado <pp.pente...@gmail.com> wrote:
> On May 11, 4:23 pm, David Fanning <n...@idlcoyote.com> wrote:
>
>> I guess that's right. I usually check to see if parameters
>> are undefined or not. In this case, because the actual programs
>> are wrappers to the PLOT command, I wanted to know how many
>> there were. Odd that I haven't run into this before, though. :-)
>
> Do you DG's plot (what I would guess by 'command')?
>
> I ask because I know there are some peculiar pitfalls in NG's plot():
> instead of (what I expected) every class inheriting Graphic, and being
> able to inherit from them, in the usual way, it is a very convoluted
> system. I suspect the cause was to make it work while minimizing the
> changes or additions to the way the iTools were organized.
>
> The most confusing part is how the classes (like Plot) are created:
> The functions (like plot()) are not the usual init functions, they are
> separate functions that create a Graphic object (itself not created by
> a init function), informing it of what kind of graphic to make. The
> graphic() function creates the proper iTool, and and an object of the
> proper class, which is just a wrapper and contains the Graphic object
> inside.
>
> This is why I gave up on trying to inherit Window for a class, and
> contained it instead (you can see it
athttp://www.ppenteado.net/idl/pp_lib/doc/pp_multiplot__defi ne.html). I
> do not remember whether it would be impossible or just complicated to
> inherit without changing the code in some of IDL's routines (which
> would be a very weird inheritance, if I had to change other code).

One way these complications manifest is that

p=plot(whatever)

does not result in the same as

p=obj_new('Plot',whatever)

as one might expect.
