Subject: Re: Object programming with data...
Posted by David Fanning on Sat, 18 May 2002 14:39:45 GMT
View Forum Message <> Reply to Message

Randall Skelton (rhskelto@atm.ox.ac.uk) writes:

> I think I have found a bug in the IDL object programming interface which
> may help explain why there are no IDL object programming books. While IDL
> claims to support methods, there appears to be no mechanism for operator
> overloading!?!! Pardon my ignorance, but in a scientific programming
> language, what is the point supporting data encapsulation without operator
> overloading?
> ...
> We can talk about Methods, polymorphism, inheritance and persistence until
> we are all blue in the face, but without the ability to define operator
> methods the usefulness of IDL in programming with data is rather limited.

I can't speak for others, but one reason I haven't written a object
programming book because it is damn hard to write a book, and too easy
to play tennis and enjoy my marriage. Another reason is that I just
keep plowing on, finding all kinds of reasons to like objects,
amazed at their unbelievable usefulness, and completely unaware of
these glaring limitations. Imagine how embarrassed I would be to write
a book that didn't even mention operator overloading! I would be the
laughing stock of the OO community. :-(

Cheers,

David
--
David W. Fanning, Ph.D.
Fanning Software Consulting
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: Object programming with data...
Posted by Randall Skelton on Sun, 19 May 2002 20:12:37 GMT
View Forum Message <> Reply to Message

On Sat, 18 May 2002, David Fanning wrote:

> I can't speak for others, but one reason I haven't written a object
> programming book because it is damn hard to write a book, and too easy
> to play tennis and enjoy my marriage.

I know this all too well... one thesis down, one more to go...

> Another reason is that I just
> keep plowing on, finding all kinds of reasons to like objects,
> amazed at their unbelievable usefulness, and completely unaware of
> these glaring limitations.

There aren't that many glaring limitations... but I think this is
certainly one of them.

> Imagine how embarrassed I would be to write
> a book that didn't even mention operator overloading! I would be the
> laughing stock of the OO community. :-(

I cannot imagine you being the laughing stock of the OO or IDL community.
To many of us, you are the mythical guru who lives high in the Colorado
mountains... I personally plan to organize a pilgrimage.

On a slightly different topic, is it possible to define a function that
takes an arbitrary number of parameters? i.e. how do I write a function
'sum' that takes 'n' variables and sums them? (yes, in this case I could
use 'total' but that's not the point...)

Cheers,
Randall

Subject: Re: Object programming with data...
Posted by Craig Markwardt on Sun, 19 May 2002 21:12:38 GMT
View Forum Message <> Reply to Message

Randall Skelton <rhskelto@atm.ox.ac.uk> writes:
> On a slightly different topic, is it possible to define a function that
> takes an arbitrary number of parameters? i.e. how do I write a function
> 'sum' that takes 'n' variables and sums them? (yes, in this case I could
> use 'total' but that's not the point...)

No fair slipping this at the end of an unrelated post! I usually
don't read David's "gosh golly" articles :-)

The answer to your question is no, and yes. No, there is no construct
in IDL that makes handling an arbitrary number of arguments easy. On
the other hand, yes, it is possible to parse them if you specify all
the parameters explicitly, as in,

```
PRO MYTOTAL, X1, X2, X3, X4, X5, X6, X7, X8, X9, X10
```

and so on up to the maximum of 64 (?). Then you access them using the
EXECUTE() function.

```
 for i = 0, n_params()-1 do begin
   dummy = execute('x = x + x'+strtrim(i,2))
 endfor
```

Craig

--

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

Subject: Re: Object programming with data...
Posted by Randall Skelton on Sun, 19 May 2002 21:22:02 GMT
View Forum Message <> Reply to Message

In response to some of my own questions I found a post from Ken Knighton in 1996 that demonstrates one way of emulating operator overloading using IDL functions... I'm open to other suggestions.

Cheers,
Randall

-- From way, way back in 1996 --

> The answer to this will have to be on the installment plan. :-)
>
> 1) Polymorphism
>
> a. Functions/procedures can be called with a variable number of
> formal parameters.
>
> b. Since identifiers are dynamically typed, a single func/pro
> can be devised that performs an operation on a variety of
> input argument types.
>
> The following tiny function shows how, by virtue of the fact that
> IDL is dynamically typed, functions can be designed with varying
> types and numbers of parameters. Note that type checking could
> be added to this function to produce errors if incompatible data
> types were used. Or, one could use the CATCH statement to react
> to any errors that may occur (such as failure to convert a string
> to a number if mixed strings and numbers were being used).
>
> ;Trivial, contrived, and useless example of "polymorphism" in IDL.

```
>  FUNCTION Add, p1, p2, p3, p4, p5, p6, p7, p9, p10
>
>    lParams = N_PARAMS()
>
>    CASE lParams OF
>
>      2L:   xSum = p1+p2
>      3L:   xSum = p1+p2+p3
>      4L:   xSum = p1+p2+p3+p4
>      5L:   xSum = p1+p2+p3+p4+p5
>      6L:   xSum = p1+p2+p3+p4+p5+p6
>      7L:   xSum = p1+p2+p3+p4+p5+p6+p7
>      8L:   xSum = p1+p2+p3+p4+p5+p6+p7+p8
>      9L:   xSum = p1+p2+p3+p4+p5+p6+p7+p8+p9
>      10L:  xSum = p1+p2+p3+p4+p5+p6+p7+p8+p9+p10
>
>      ELSE: MESSAGE, 'Must use 2 through 10 parameters.'
>    ENDCASE
>
>    RETURN, xSum
>  END
>
>  There are also ways of doing the above without using a CASE statement.
>  One of these is to use the EXECUTE command and a FOR loop:
>
>  xSum = p1+p2
>  FOR i=3, lParams DO BEGIN
>    aExec = 'xSum = xSum + p'+STRTRIM(i,2)
>    lErr = EXECUTE(aExec)
>  ENDFOR
>
>  Of course, the case statement runs much more quickly and is more
>  obvious in its logic.  However, the EXECUTE statement has its place
>  and provides on-the-fly compilation and execution of statements.
>
>  If you call the above function using a variety of input types, you will
>  soon notice that the actual parameters can be of any numeric or string
>  type and can be either scalars or arrays.  If strings and numerics are
>  mixed, then the strings must be able to convert to numeric type.  One
>  can not use structures in the above example, but one could modify this
>  code to check for structures using the SIZE function and then take
>  action accordingly.
>
>  As you can see, it is fairly easy to write one function that takes
>  care of a wide variety of possibilities for input arguments.
>
>  I'll try to continue this discussion later.  Any feedback is welcome.
>  If someone has a better example, please post.
```

>
> Ken Knighton knighton@gav.gat.com knighton@cts.com
> General Atomics
> San Diego, CA
>

Subject: Re: Object programming with data...
Posted by David Fanning on Sun, 19 May 2002 22:20:04 GMT
View Forum Message <> Reply to Message

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

> No fair slipping this at the end of an unrelated post! I usually
> don't read David's "gosh golly" articles :-)

What!? Those are the only ones that make any sense. :-)

Cheers,

David

P.S. It has come to my attention (once again) that
humor (and in particular sarcasm) doesn't translate
well into the Asiatic language group. So just for the
record, I'm not mad at anyone, and especially not at
Craig. :-)

--
David W. Fanning, Ph.D.
Fanning Software Consulting
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: Object programming with data...
Posted by David Fanning on Sun, 19 May 2002 22:28:25 GMT
View Forum Message <> Reply to Message

Randall Skelton (rhskelto@atm.ox.ac.uk) writes:

> I cannot imagine you being the laughing stock of the OO or IDL community.
> To many of us, you are the mythical guru who lives high in the Colorado
> mountains... I personally plan to organize a pilgrimage.

I wouldn't do that. Others have done it and been universally

disappointed. Ask Pavel. :-)

Cheers,

David

P.S. Let's just say I'm a hell of a lot more fun on
paper than I am in person. :-(

--
David W. Fanning, Ph.D.
Fanning Software Consulting
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: Object programming with data...
Posted by Jaco van Gorkom on Wed, 22 May 2002 15:39:12 GMT
View Forum Message <> Reply to Message

"Randall Skelton" <rhskelto@atm.ox.ac.uk> wrote in message
news:Pine.LNX.4.33.0205192219030.712-100000@moriarty.atm.ox. ac.uk...
> In response to some of my own questions I found a post from Ken Knighton
> in 1996 that demonstrates one way of emulating operator overloading using
> IDL functions... I'm open to other suggestions.

I have used CASE statements in the past, but also occasionally made use of
recursion, something like:

```
function Add, p1, p2, p3, p4, p5, p6
  if n_elements(p3) eq 0 then $
   return, p1+p2 $
  else $
   return, Add(p1+p2, p3, p4, p5, p6)
end
```

There is a variant which can be very quick (and sort of ugly) to add to the
start of an existing function:

```
function Add, p1, p2, p3, p4, p5, p6
  ; Add this statement for recursive parameter handling
  if n_elements(p3) ne 0 then $
   return, Add( Add(p1,p2), p3, p4, p5, p6)
  ; Original two-parameter function code follows:
  return, p1+p2
end
```

Cheers,
Jaco

Subject: Re: Object programming with data...
Posted by merlecorp on Wed, 22 May 2002 21:52:38 GMT
View Forum Message <> Reply to Message

Craig Markwardt <craigmnet@cow.physics.wisc.edu> wrote in message
news:<onptzrj0zd.fsf@cow.physics.wisc.edu>...
> Randall Skelton <rhskelto@atm.ox.ac.uk> writes:
>> On a slightly different topic, is it possible to define a function that
>> takes an arbitrary number of parameters? i.e. how do I write a function
>> 'sum' that takes 'n' variables and sums them? (yes, in this case I could
>> use 'total' but that's not the point...)
>
> No fair slipping this at the end of an unrelated post! I usually
> don't read David's "gosh golly" articles :-)
>
> The answer to your question is no, and yes. No, there is no construct
> in IDL that makes handling an arbitrary number of arguments easy. On
> the other hand, yes, it is possible to parse them if you specify all
> the parameters explicitly, as in,
>
> PRO MYTOTAL, X1, X2, X3, X4, X5, X6, X7, X8, X9, X10
>
> and so on up to the maximum of 64 (?). Then you access them using the
> EXECUTE() function.
>
> for i = 0, n_params()-1 do begin
> dummy = execute('x = x + x'+strtrim(i,2))
> endfor
>
>
> Craig

Hey,

Here's another approach using the _extra keyword. It's not my idea--I
pilfered it from R. Kling's InformationPanel--but I liked his approach
to arbitrary inputs & thought it was worth posting in this simplified
form.

One catch ... I'm not sure if there are limits on the number of
parameters accepted by the _extra keyword.

cheers,

merle

```
;  ------------------------------------------------------------ -------
FUNCTION Sum, _extra=extra
; -- borrowed this approach from R. Kling's InformationPanel.pro

nNumEntries = N_Tags(extra)

nSum = 0
FOR i = 0, nNumEntries-1 DO nSum = nSum + extra.(i)

Return, nSum

END


;  ------------------------------------------------------------ -------
PRO eg
; -- example calling program

x1 = 10
x2 = 20
x3 = 30
x4 = 40
x5 = 50

Print, Sum(x1=x1, x2=x2, x3=x3)
Print, Sum(x1=x1, x2=x2, x3=x3, x4=x4)
Print, Sum(x1=x1, x2=x2, x3=x3, x4=x4, x5=x5)

END
```

Subject: Re: Object programming with data...
Posted by Pavel A. Romashkin on Thu, 23 May 2002 15:28:01 GMT
View Forum Message <> Reply to Message

David Fanning wrote:
>
> I wouldn't do that. Others have done it and been universally
> disappointed. Ask Pavel. :-)
>
> P.S. Let's just say I'm a hell of a lot more fun on
> paper than I am in person. :-(

David must be tired from playing tennis, or else he'd never post this. I think he is the nicest person you can meet. One thing I *am*

disappointed with is that David apparently lost interest in microbrews and I have not found another lure for him yet. IDL as a lure? You are kidding, right?
So, this is gotta be me who is to be avoided. Must be my English :-(
Banished to the Mojave Desert,
Pavel
