Subject: IDL 8.4?
Posted by rryan%stsci.edu on Wed, 15 Oct 2014 14:55:09 GMT
View Forum Message <> Reply to Message

I stumbled onto the "Whats's New in 8.4" page this morning...

http://www.exelisvis.com/docs/WhatsNew.html

I was under the notion that 8.4 wasn't coming out until November, but I guess it's out early. Anywho. It looks like they're really moving toward object-oriented techniques and a very pythonic experience, so I wanted to know if anyone has the inside track on some things:

(1) what is this lambda-inline function business? How is it different than execute()? Can it be used with the VM? What advantages does it have?

(2) they're adding static methods to all variables? does this mean variables must be deleted (a la obj_destroy)? Can variables still change types (like x=0 & x=x+2.0 & help,x)? What happens when I evoke a string-based method on a numerical type? How does the execution time for these new static methods compare to the "functional" or "procedural" versions? I guess, what advantage do you add here, other than complete loss of backwards compatibility (or forward, I don't know. I just know I can't use these methods and give to a friend who's still running <8.4).

(3) Thye've added a boolean datatype. Is it really boolean, or is it just a byte or int that is capped at 1? I mean, will intrinsically boolean things (which should've returned booleans from day 1 which return ints such as: x = 1 lt 2 or t=keyword_set(SOME_VARIABLE)) return booleans now? So now you could do use keyword_set() to test if a keyword is set versus using keyword_set to see if the keyword is set to something non-zero (like it's always been).

I've got more questions, but these are the most troubling.

Russell

Subject: Re: IDL 8.4?
Posted by chris_torrence@NOSPAM on Wed, 15 Oct 2014 16:01:16 GMT
View Forum Message <> Reply to Message

On Wednesday, October 15, 2014 8:55:10 AM UTC-6, rryan%s...@gtempaccount.com wrote:
> I stumbled onto the "Whats's New in 8.4" page this morning...
>
>
>
> http://www.exelisvis.com/docs/WhatsNew.html
>
>
>
> I was under the notion that 8.4 wasn't coming out until November, but I guess it's out early. Anywho. It looks like they're really moving toward object-oriented techniques and a very pythonic

experience, so I wanted to know if anyone has the inside track on some things:
>
>
>
> (1) what is this lambda-inline function business? How is it different than execute()? Can it be used with the VM? What advantages does it have?
>
>
>
> (2) they're adding static methods to all variables? does this mean variables must be deleted (a la obj_destroy)? Can variables still change types (like x=0 & x=x+2.0 & help,x)? What happens when I evoke a string-based method on a numerical type? How does the execution time for these new static methods compare to the "functional" or "procedural" versions? I guess, what advantage do you add here, other than complete loss of backwards compatibility (or forward, I don't know. I just know I can't use these methods and give to a friend who's still running <8.4).
>
>
>
> (3) Thye've added a boolean datatype. Is it really boolean, or is it just a byte or int that is capped at 1? I mean, will intrinsically boolean things (which should've returned booleans from day 1 which return ints such as: x = 1 lt 2 or t=keyword_set(SOME_VARIABLE)) return booleans now? So now you could do use keyword_set() to test if a keyword is set versus using keyword_set to see if the keyword is set to something non-zero (like it's always been).
>
>
>
> I've got more questions, but these are the most troubling.
>
>
>
> Russell

Hi Russell,

You beat me to the announcement. I just posted a long blurb about IDL 8.4 if you want to read about the new features.

Regarding your specific concerns, I would recommend downloading IDL 8.4 and playing around with the new features. In short, it's all "good" stuff - we haven't broken anything, and nothing has changed which would break backwards compatibility.

As is always true, if you use new features, those won't be compatible with older versions of IDL. So if you are developing a library for other people, you certainly want to be careful.

Regarding your specific questions about "static methods on variables" - IDL variables still work exactly like they did before. You can still change types, you don't need to clean them up, etc. The new methods should be the same speed or perhaps just a tiny bit slower (there is a small overhead for calling the method). If you are calling a method a million times in a loop you might

notice, but for the normal IDL case (making one call with a million elements) you won't even notice.

Overall, our development goals are:
1. Keep existing users happy by adding feature requests and fixing bugs.
2. Move the language forward (yes, "like python") to attract new users.

Cheers,
Chris
ExelisVIS

Subject: Re: IDL 8.4?
Posted by Fabzi on Wed, 15 Oct 2014 16:03:17 GMT
View Forum Message <> Reply to Message

On 15.10.2014 16:55, rryan%stsci.edu@gtempaccount.com wrote:
> I stumbled onto the "Whats's New in 8.4" page this morning...
>
> http://www.exelisvis.com/docs/WhatsNew.html

Thanks for the link! Indeed very pythonic... I personally think these are great improvements.

> (1) what is this lambda-inline function business?

I'm not sure about these lambda functions. In python a lot of stuff is based on iterators and in this case I understand why it's useful, but in IDL I dunnow...

> (2) they're adding static methods to all variables?

obj_destroy is "obsolete" since IDL8. I personally think these methods make code MUCH funnier to write and easy to read than the cumbersome use of SIZE(). Which operator overloading, this will encourage versatile code based on duck typing.
I don't really understand the problem of backward compatibility you mention, since this has always been like this for each IDL version. One can still write code without list() and hash(), but then what's the point of buying IDL8?

More generally, do you think that Exelis should invest more time somewhere else? Sure, they could develop new tools for the standard library but they will never be able to compete with python's huge user base, so I think that it's great that they make the language more flexible and funny to code with.

Fabien

Subject: Re: IDL 8.4?
Posted by rryan%stsci.edu on Wed, 15 Oct 2014 16:44:12 GMT
View Forum Message <> Reply to Message

Hi Chris and Fabien

Thanks for the tips. I wasn't criticizing anything, even the choice of moving more OO'ed. Over the years I've grown to appreciate, even prefer objects. I guess I remain bit reluctant to go full object (in IDL) because of limitations with passing objects into IDL_IDLBridge or clunkiness with saving/restoring them and so on. But that's a minor issue.

I understand the issue with compatability --- If i develop something using a modern tool (such as a hash), then it's never going to work on some older version. But that wasn't my issue. I was asking a bit more rhetorically, what do I gain with these new techniques (such as static methods)? Because I can see what is lost, but I can't see what is gained. This is not to say that nothing is gained, I just was a bit unclear what that was exactly. I can understand that the static methods case is maybe a bit pedantic, because it's a low-level addition designed to facilitate higher-level operations after all the choice to do

sz= size(var)

vs.

sz= var.size()

is really just a matter of preference and syntax, not one of efficiency or so on (As a note, remember even python has the len() function for this purpose and while it's heralded as a object-oriented many things are still very functional --- which has always annoyed me.)

I really like the changes and do see them as upgrades. But I just wanted clarification on the upgrades, as I often do work in modest collaborations where we share code and so on. Because if there's clear advantage to certain things (as opposed to conceptual reorganization of existing tools), I want to know about it and encourage co-Is to upgrade from IDL 7.x. That's all I was getting at. Again, I like the more OO'ed nature, for many problems OO is really a superior mindset (graphics, widgets, come to mind).

All the best gang,
Russell

On Wednesday, October 15, 2014 12:03:20 PM UTC-4, Fabien wrote:
> On 15.10.2014 16:55, rryan%stsci.edu@gtempaccount.com wrote:
>
>> I stumbled onto the "Whats's New in 8.4" page this morning...

>
>>
>
>> http://www.exelisvis.com/docs/WhatsNew.html
>
>
>
> Thanks for the link! Indeed very pythonic... I personally think these
>
> are great improvements.
>
>
>
>> (1) what is this lambda-inline function business?
>
>
>
> I'm not sure about these lambda functions. In python a lot of stuff is
>
> based on iterators and in this case I understand why it's useful, but in
>
> IDL I dunnow...
>
>
>
>> (2) they're adding static methods to all variables?
>
>
>
> obj_destroy is "obsolete" since IDL8. I personally think these methods
>
> make code MUCH funnier to write and easy to read than the cumbersome use
>
> of SIZE(). Which operator overloading, this will encourage versatile
>
> code based on duck typing.
>
> I don't really understand the problem of backward compatibility you
>
> mention, since this has always been like this for each IDL version. One
>
> can still write code without list() and hash(), but then what's the
>
> point of buying IDL8?
>
>
>
>

>
> More generally, do you think that Exelis should invest more time
>
> somewhere else? Sure, they could develop new tools for the standard
>
> library but they will never be able to compete with python's huge user
>
> base, so I think that it's great that they make the language more
>
> flexible and funny to code with.
>
>
>
> Fabien

Subject: Re: IDL 8.4?
Posted by chris_torrence@NOSPAM on Wed, 15 Oct 2014 17:01:13 GMT
View Forum Message <> Reply to Message

On Wednesday, October 15, 2014 10:44:14 AM UTC-6, rryan%s...@gtempaccount.com wrote:
> Hi Chris and Fabien
>
>
>
> Thanks for the tips. I wasn't criticizing anything, even the choice of moving more OO'ed. Over the years I've grown to appreciate, even prefer objects. I guess I remain bit reluctant to go full object (in IDL) because of limitations with passing objects into IDL_IDLBridge or clunkiness with saving/restoring them and so on. But that's a minor issue.
>
>
>
> I understand the issue with compatability --- If i develop something using a modern tool (such as a hash), then it's never going to work on some older version. But that wasn't my issue. I was asking a bit more rhetorically, what do I gain with these new techniques (such as static methods)? Because I can see what is lost, but I can't see what is gained. This is not to say that nothing is gained, I just was a bit unclear what that was exactly. I can understand that the static methods case is maybe a bit pedantic, because it's a low-level addition designed to facilitate higher-level operations after all the choice to do
>
>
>
> sz= size(var)
>
>
>
> vs.
>

>
>
> sz= var.size()
>
>
>
> is really just a matter of preference and syntax, not one of efficiency or so on (As a note, remember even python has the len() function for this purpose and while it's heralded as a object-oriented many things are still very functional --- which has always annoyed me.)
>
>
>
> I really like the changes and do see them as upgrades. But I just wanted clarification on the upgrades, as I often do work in modest collaborations where we share code and so on. Because if there's clear advantage to certain things (as opposed to conceptual reorganization of existing tools), I want to know about it and encourage co-Is to upgrade from IDL 7.x. That's all I was getting at. Again, I like the more OO'ed nature, for many problems OO is really a superior mindset (graphics, widgets, come to mind).
>
>
>
> All the best gang,
>
> Russell
>
>
>

Hi Russell,

No worries. Sometimes I get a bit defensive (for understandable reasons).

I'll give just two examples of using Lambda that might be interesting:

```
result = QROMB(LAMBDA(x:x^3 + (x-1)^2 + 3), -4, 4)

p = PLOT(Lambda(x:x^3 + (x-1)^2 + 3), XRANGE=[-4,4], $
  /FILL_BACKGROUND, FILL_LEVEL=0)
```

Finally, as an exercise for the reader: what about using a Lambda as a widget event function (say for a widget_button)?

-Chris