Subject: Re: Namespaces (redux)
Posted by R.Bauer on Wed, 25 Sep 2002 07:55:59 GMT
View Forum Message <> Reply to Message

JD Smith wrote:
> I wrote this almost two weeks ago in another thread, but the newfeed here
> seems to have taken a stance resolutely against free expression. They tell
> me the problem is fixed now. I figure it's a good test because it's a
> controversial enough topic that I'll get immediate responses, if it makes
> it out:
>
>
> On Wed, 11 Sep 2002 23:42:22 -0700, Reimar Bauer wrote:
>
>
>> David Fanning wrote:
>>
>>> Wayne Landsman (landsman@mpb.gsfc.nasa.gov) writes:
>>>
>>>
>>>
>>>> I have heard a rumor that there may be a standardized way of counting
>>>> the number of lines in a file in the next release of IDL ;-)
>>>
>>>
>>> And I hear it even has a name very similar to Riemer's little File_Line
>>> program, which I think is too bad. Something like COUNT_ROWS really
>>> makes more sense to me. :-)
>>>
>>> Cheers,
>>>
>>> David
>>
>>
>> I don't know at the moment if I should be happy or not. It's fine to see
>> that's good routines would be implemented into the idl binary but always
>> this is done I got the problem that's all of our sources using these
>> routines need changes. This happens last time by file_search. We have
>> had nearly the same functionality in our routine but not the same
>> parameters or keywords. Internal routines are first called sources with
>> the same name are ignored.
>>
>> I believe they like to start with FILE_ because of the other file
>> handling routines. I would prefer FILE_COUNT_ROWS if possible. This
>> gives more sense as the word I have choosen in the past.
>
>
> This brings up an excellent point that probably needs reiterating: the IDL

> namespace is a rapidly dwindling resource, as libraries grow and
> disseminate. While in the past most IDL users might have worked primarily
> with their own routines, with a few bits and pieces of outside code
> cobbled together from various sources, that era seems to be passing, given
> the improving quality and availability of large libraries, and the new RSI
> initiative to facilitate code sharing among IDL users.

This is fine for some people who haven't access to html servers to put
some projects there, but I like also to add only the URL too.
At the moment I am waiting a bit how it goes on.

Some time ago as dejanews has gone we have had a lot of trouble because
there was no access to the archive. If now the experience of this group
is splitted into rsinc forum and idl-pvwave we will lose again much of
the results of discussions and ideas.
I don't believe that's the rsinc forum is accesible by google forum
search or am I wrong.

> What can we do to
> preserve this vanishing resource? A few guidelines come to mind:
>
> 1. Use Objects. Object methods don't count against the namespace,
> since they're encapsulated beneath the class name. Use unique class
> names. "Box" is probably not a good name for a class.
> "tvRubberBandBox" might be better.

Later on for some new routines this is possible. But what to do with the
actual lib.

>
> 2. If not using objects, imitate their namespace parsimony by
> consistently prepending a class-like prefix to all related routines...
> e.g. tvRubberBandBoxDisplay.pro, or tvRBandBox_display.

uppercases in routine names arent't accesible by unix or linux.
Each of these routines must be compiled first.
Who has asked for this feature before, if no one I will do it.

This changing is more complicated as the changes rsinc did.

In the past it was easy to add a new library for me because there
weren't much available for athmospheric science. But at the moment it is
difficult by source. By idl binary it isn't. This was one of the reasons
why I have written my compile routine.
The whole plot (plotxy,plot2d ..) environment of our library is stored
in the plotprepare.sav file. And by using the initialising script
plotprepare,plot this is loaded at once. After loading the sav file

you have access to an info function x=plotprepare_info() where you can get informations about the sources. All sources are available in the catalog so you don't miss something.
I added to the sav file a lifetime of about I believe one month, because if you are working on a library the functionality get more improvements.

At home this is no problem because it could be recompiled each day. I will add soon a mechanism to our web page where actual plotprepare.sav files could be restored.

To load an additional library without changes means changing the search path.
Normally it should be constant to get the same results.
It could be changed during a session but I don't know how to remove a compiled routine from memory. This is possible because idlde is able to do it if I delete a first edited and compiled routine. I won't removing all from memory.

>
> 3. Resist the urge to give routines short catchy names, especially if
> they will ever be distributed (even if just emailed to a colleague).
> To save time for interactive use, consider using local "abbreviation"
> routines which will never be distributed, and which call longer-winded
> programs, e.g.
>
> pro tvrbd, args, _EXTRA=e
> tvRubberBandBoxDisplay, args, _EXTRA=e
> end
>
> Never call these interactive abbreviation routines directly in any
> code.

This don't solve the problem, then the filenames and routines are doubled.

>
> 4. Make qualifying prefixes descriptive and unique. Using initials
> as qualifiers is of course an option, but wastes precious filename
> space on information of little real value. If you do use initials,
> don't skimp on additional information in the rest of the name.

We use this sometimes and sometimes not. It's more used to categorize routines.

>
> 5. Look before you leap. An excellent (if increasingly outdated)
> online-browser for almost all widely distributed libraries of IDL code
> is available at:

>
> http://www.astro.washington.edu/deutsch/idl/htmlhelp/
>
>   Use it to help you pick unique names, but don't imitate the
>   spendthrift ways of our forerunners: we've inherited the problem from
>   them, and must act to counter it.

I thinks it is easier if we can share idlwave_catalog.el files with the
library routines.

>
> 6. If you are internalizing and distributing individual routines from
>   libraries, with no plan to maintain compatibility with future versions
>   of that code, change the name. If your code calls a routine with an
>   established name (e.g., readfits), it should endeavor to work with the
>   latest version of that routine. Otherwise, the name should be changed
>   (e.g. projx_readfits). At the very least, document what version of
>   the code it requires.
>
> Other suggestions or additions?
>
> JD

--
Reimar Bauer

Institut fuer Stratosphaerische Chemie (ICG-I)
Forschungszentrum Juelich
email: R.Bauer@fz-juelich.de
 -- -------
        a IDL library at ForschungsZentrum Juelich
  http://www.fz-juelich.de/icg/icg-i/idl_icglib/idl_lib_intro. html
 == =======

Subject: Re: Namespaces (redux)
Posted by Struan Gray on Wed, 25 Sep 2002 08:42:07 GMT
View Forum Message <> Reply to Message

JD Smith, jdsmith@as.arizona.edu writes:

> 1. Use Objects.

 I think this is the key. If I were ever to put together a library
of routines suitable for public use (oink oink, flap, flap) the most
natural way seems to create a package object and make all the routines
methods. There is no law that says an object's methods have to act on

its internal data. Converting existing libraries and the code that uses it could probably even be automated.

Xwindow gets my vote for the highest combination of usefulness and namespace conflicts (you know who you are....).

Struan

Subject: Re: Namespaces (redux)
Posted by David Fanning on Wed, 25 Sep 2002 13:56:36 GMT
View Forum Message <> Reply to Message

JD Smith (jdsmith@as.arizona.edu) writes:

> Other suggestions or additions?

I support JD's suggestions completely, especially Number 2:

> 2. If not using objects, imitate their namespace parsimony by
>   consistently prepending a class-like prefix to all related routines...
>   e.g. tvRubberBandBoxDisplay.pro, or tvRBandBox_display.

You can't believe how many widget programs I see where this practice is not followed. How many QUIT procedures can we have in this world? It's remarkable that these programs ever work at all.

I'd make just one other suggestion: sleep on your naming decision overnight before you make it public. Once it is out there, it is pretty much out there forever. There is no going back, ever, which can lead to all kinds of embarrassment and extra 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: Namespaces (redux)
Posted by Craig Markwardt on Wed, 25 Sep 2002 15:06:00 GMT
View Forum Message <> Reply to Message

JD Smith <jdsmith@as.arizona.edu> writes:
> This brings up an excellent point that probably needs reiterating: the IDL
> namespace is a rapidly dwindling resource, as libraries grow and
> disseminate. While in the past most IDL users might have worked primarily
> with their own routines, with a few bits and pieces of outside code
> cobbled together from various sources, that era seems to be passing, given
> the improving quality and availability of large libraries, and the new RSI
> initiative to facilitate code sharing among IDL users. What can we do to
> preserve this vanishing resource? A few guidelines come to mind:

JD, I think you are being a bit too alarmist. The IDL namespace is not rapidly dwindling the same way that (pick favorite species) are declining. For a typical 10-character function name, there are about a million billion unique combinations. Of course, if we restrict to English-sounding names, then the unique combinations is less, but still huge.

Still, I agree with several of your points, just at a philosophical level:

> 2. If not using objects, imitate their namespace parsimony by
> consistently prepending a class-like prefix to all related routines...
> e.g. tvRubberBandBoxDisplay.pro, or tvRBandBox_display.

Yes, this is a must, at least for routines in the same file.

> 4. Make qualifying prefixes descriptive and unique. Using initials
> as qualifiers is of course an option, but wastes precious filename
> space on information of little real value. If you do use initials,
> don't skimp on additional information in the rest of the name.

I'm not sure it's of little real value. For example, from the names REPLICATE and CMREPLICATE, you can probably guess that both do something similar.

Or, the names of functions MPFIT, MPFITFUN, MPFITEXPR, MPFITPEAK, and so on, indicate that they all belong to the same library, obey the same calling conventions, and that is valuable knowledge. [And pays homage to the MINPACK heritage.]

I'm not a huge fan of typing routine names like tvRubberBandBoxUpdatePanelWithMarchingAnts().

> 5. Look before you leap. An excellent (if increasingly outdated)
> online-browser for almost all widely distributed libraries of IDL code
> is available at:

Now here is the crux of the matter. The reason that we have name clashes is that everybody and their brother has written *slightly different* routines to count lines in files, called it the same thing, and published it. However, most people are respectful. If you search the IDL Astronomy library or the U. Washington index for terms like FIT or LEGENDRE, you will find lots of routines, but they are for the most part unique. Rule: If it's an *obvious* name, then attach a *non* obvious prefix or suffix to the name. End of story.

It is notable that in almost every case, the offending entity for these types of namespace clashes is none other than Research Systems Incorporated. We should be preaching to them.

Yours,
Craig

My addition: I attempt to always keep the main routine name unique within the first 8 characters from all other routines I know about.

Subject: Re: Namespaces (redux)
Posted by JD Smith on Wed, 25 Sep 2002 15:32:27 GMT
View Forum Message <> Reply to Message

On Wed, 25 Sep 2002 00:55:59 -0700, Reimar Bauer wrote:

> JD Smith wrote:
> This is fine for some people who haven't access to html servers to put
> some projects there, but I like also to add only the URL too. At the
> moment I am waiting a bit how it goes on.
>
> Some time ago as dejanews has gone we have had a lot of trouble because
> there was no access to the archive. If now the experience of this group
> is splitted into rsinc forum and idl-pvwave we will lose again much of
> the results of discussions and ideas. I don't believe that's the rsinc
> forum is accesible by google forum search or am I wrong.
>
>
>> What can we do to
>> preserve this vanishing resource? A few guidelines come to mind:
>>
>> 1. Use Objects. Object methods don't count against the namespace,

>> since they're encapsulated beneath the class name. Use unique
>> class names. "Box" is probably not a good name for a class.
>> "tvRubberBandBox" might be better.
>
> Later on for some new routines this is possible. But what to do with the
> actual lib.
>
>
>> 2. If not using objects, imitate their namespace parsimony by
>> consistently prepending a class-like prefix to all related
>> routines...
>> e.g. tvRubberBandBoxDisplay.pro, or tvRBandBox_display.
>
> uppercases in routine names arent't accesible by unix or linux. Each of
> these routines must be compiled first. Who has asked for this feature
> before, if no one I will do it.
>
> This changing is more complicated as the changes rsinc did.
>
> In the past it was easy to add a new library for me because there
> weren't much available for athmospheric science. But at the moment it is
> difficult by source. By idl binary it isn't. This was one of the reasons
> why I have written my compile routine. The whole plot (plotxy,plot2d ..)
> environment of our library is stored in the plotprepare.sav file. And by
> using the initialising script plotprepare,plot this is loaded at once.
> After loading the sav file you have access to an info function
> x=plotprepare_info() where you can get informations about the sources.
> All sources are available in the catalog so you don't miss something. I
> added to the sav file a lifetime of about I believe one month, because
> if you are working on a library the functionality get more
> improvements.

This of course doesn't really solve the problem at all, but merely circumvents it in a confining way. By creating a SAV file of your code, you've just found a clever way to put your routines in front of other routines on the !PATH. The routine shadowing still exists, but you've guaranteed yours comes out on top.

What if I have *another* plotxy command from another library (or one I created myself)? After I load your binary, that will no longer be accessible. The idea is to "preserve", not "capture" the namespace. Another route along these lines many packages have gone is to provide a shell script which sets or modifies !PATH before running IDL with the custom package. This provides a similar namespace "capture", but suffers from an even worse form of the same problem: now I can't use *any* of my other routines from other libraries when running the package!

>> 3. Resist the urge to give routines short catchy names, especially if
>> they will ever be distributed (even if just emailed to a
>> colleague). To save time for interactive use, consider using local
>> "abbreviation" routines which will never be distributed, and which
>> call longer-winded programs, e.g.
>>
>> pro tvrbd, args, _EXTRA=e
>> tvRubberBandBoxDisplay, args, _EXTRA=e
>> end
>>
>> Never call these interactive abbreviation routines directly in any
>> code.
>
> This don't solve the problem, then the filenames and routines are
> doubled.

It solves the problem of users who are motivated, by typing laziness, to pick very short, and thus incredibly wasteful to the namespace, routine names. I for one never use this, because IDLWAVE completes my routine names for me, usually after just a few keystrokes (yes, even for object methods).

>> 5. Look before you leap. An excellent (if increasingly outdated)
>> online-browser for almost all widely distributed libraries of IDL
>> code is available at:
>>
>> http://www.astro.washington.edu/deutsch/idl/htmlhelp/
>>
>> Use it to help you pick unique names, but don't imitate the
>> spendthrift ways of our forerunners: we've inherited the problem
>> from them, and must act to counter it.
>
> I thinks it is easier if we can share idlwave_catalog.el files with the
> library routines.

Reimar is referring to a system I've proposed for IDLWAVE whereby widely-distributed libraries would have a catalog prescanned and included with the distribution. Of course, IDL will still do nothing about it if you have shadowed routines on your !PATH. IDLWAVE can at least alert you to this problem, but can't solve it (you have to compile one manually, rename it, etc.... gets to be a pain if it's in a constantly updated library directory).

Glad to be back from nameserver exile.

JD

Subject: Re: Namespaces (redux)
Posted by JD Smith on Wed, 25 Sep 2002 17:13:46 GMT
View Forum Message <> Reply to Message

On Wed, 25 Sep 2002 08:06:00 -0700, Craig Markwardt wrote:
> JD Smith <jdsmith@as.arizona.edu> writes:
>> This brings up an excellent point that probably needs reiterating: the
>> IDL namespace is a rapidly dwindling resource, as libraries grow and
>> disseminate. While in the past most IDL users might have worked
>> primarily with their own routines, with a few bits and pieces of
>> outside code cobbled together from various sources, that era seems to
>> be passing, given the improving quality and availability of large
>> libraries, and the new RSI initiative to facilitate code sharing among
>> IDL users. What can we do to preserve this vanishing resource? A few
>> guidelines come to mind:
>
> JD, I think you are being a bit too alarmist. The IDL namespace is not
> rapidly dwindling the same way that (pick favorite species) are
> declining. For a typical 10-character function name, there are about a
> million billion unique combinations. Of course, if we restrict to
> English-sounding names, then the unique combinations is less, but still
> huge.

The alarmism was a clever ploy to engage critical thinkers such as yourself ;). By no means am I equating the IDL namespace to our rainforests or endangered species (after all, we could always all just switch to Python: the red panda doesn't have that option).

Yes, if IDL programmers would choose randomly even from the 8^30 or so potential 8.3 DOS-friendly names, this would never be a problem. And even if they stuck to the 50,000 or so english words with 8 or fewer letters, trouble would be averted. This, as you know, is not the case. Instead, they understandably tend to cluster around a very limited set of key words like "plot", "func", etc. For instance, in my not-overly-large collection of libraries, I found 131 of 1991 total routines had the word "plot" in their name. Thirty of these were object methods, so those can be forgiven, but the extremely small portion of the potential namespace we stick to is evident: it's *much* smaller than the domain of English-sounding names.

>> 4. Make qualifying prefixes descriptive and unique. Using initials
>> as qualifiers is of course an option, but wastes precious filename
>> space on information of little real value. If you do use initials,
>> don't skimp on additional information in the rest of the name.

> I'm not sure it's of little real value. For example, from the names
> REPLICATE and CMREPLICATE, you can probably guess that both do something
> similar.
>

> Or, the names of functions MPFIT, MPFITFUN, MPFITEXPR, MPFITPEAK, and so
> on, indicate that they all belong to the same library, obey the same
> calling conventions, and that is valuable knowledge. [And pays homage
> to the MINPACK heritage.]
>
> I'm not a huge fan of typing routine names like
> tvRubberBandBoxUpdatePanelWithMarchingAnts().

Yes, you have to weigh conciseness and clearness against potential for namespace collision. But clearness and small namespace footprint are not congruent: I could also guess that "plotxy" does something similar to "plot", which makes it a descriptive name, but simultaneously gives it a high likelihood for collision. I also don't like overly long routine names: the point is to be descriptive and unique, not pedagogical. In the case of your MINPACK functions, that's a perfect example of using a clear, well-motivated prefix qualifier.

>> 5. Look before you leap. An excellent (if increasingly outdated)
>> online-browser for almost all widely distributed libraries of IDL
>> code is available at:
>
>
> Now here is the crux of the matter. The reason that we have name
> clashes is that everybody and their brother has written *slightly
> different* routines to count lines in files, called it the same thing,
> and published it. However, most people are respectful. If you search
> the IDL Astronomy library or the U. Washington index for terms like FIT
> or LEGENDRE, you will find lots of routines, but they are for the most
> part unique. Rule: If it's an *obvious* name, then attach a *non*
> obvious prefix or suffix to the name. End of story.

An excellent synopsis of the recommendation. Unfortunately, unless you've been in the business a long time, it might not be clear what an *obvious* name is. Hence the guidelines. On another note, Eric Deutsch is no longer maintaining the IDL library browser. I'm sure he'd be pleased to pass it along to another willing maintainer.

> It is notable that in almost every case, the offending entity for these
> types of namespace clashes is none other than Research Systems
> Incorporated. We should be preaching to them.

On that point I fully agree. And I have. I'm sure RSI feels a distinct and righteous ownership of the IDL namespace and believes that routines in !DIR are free to trample all over it. Here's an interesting random bit of output from IDLWAVE's shadow detection:

PNT_LINE()
- SystemLib [C--] /usr/local/rsi/idl_5.5/lib/pnt_line.pro

- SystemLib [C--] /usr/local/rsi/idl_5.5/lib/efont.pro
read_pict_item
- SystemLib [C--] /usr/local/rsi/idl_5.5/lib/read_pict.pro
- SystemLib [C--] /usr/local/rsi/idl_5.5/lib/query_pict.pro
READ_PPM_NEXT_LINE()
- SystemLib [C--] /usr/local/rsi/idl_5.5/lib/read_ppm.pro
- SystemLib [C--] /usr/local/rsi/idl_5.5/lib/query_ppm.pro
READ_PPM_NEXT_TOKEN()
- SystemLib [C--] /usr/local/rsi/idl_5.5/lib/read_ppm.pro
- SystemLib [C--] /usr/local/rsi/idl_5.5/lib/query_ppm.pro

In any case, it's too late to undo the sins of the parent, but we can strive to avoid repeating them ourselves.

> My addition: I attempt to always keep the main routine name unique
> within the first 8 characters from all other routines I know about.

A fine suggestion, if only for those few remaining fossil systems which enforce such a filename length limit.

Thanks for the comments.

JD

Subject: Re: Namespaces (redux)
Posted by Wayne Landsman on Wed, 25 Sep 2002 17:49:12 GMT
View Forum Message <> Reply to Message

>
> Other suggestions or additions?
>
> JD

A modest one is to "retire" procedures which are no longer being used because their functionality has been replaced. For example, as of V5.5, the /obsolete directory from RSI contains 147 procedures. I suppose that years must pass before one can say that these names have truly returned to the namespace -- but I notice that years seem to pass more quickly than they once did. --Wayne

---

Subject: Re: Namespaces (redux)
Posted by David Fanning on Wed, 25 Sep 2002 18:26:50 GMT
View Forum Message <> Reply to Message

Wayne Landsman (landsman@mpb.gsfc.nasa.gov) writes:

> I suppose that
> years must pass before one can say that these names have truly returned to
> the namespace -- but I notice that years seem to pass more quickly than they
> once did.

Indeed, I even wrote "1999" on a check at the bank the
other day, for some inexplicable reason. I couldn't
understand why the young lady teller was looking at
me so pitifully. I thought maybe I'd forgotten to trim
the nose hair again before I started flirting with her. :-(

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: Namespaces (redux)
Posted by R.Bauer on Wed, 25 Sep 2002 19:41:51 GMT
View Forum Message <> Reply to Message

JD Smith wrote:
> On Wed, 25 Sep 2002 00:55:59 -0700, Reimar Bauer wrote:
>
>
>> JD Smith wrote:
>> This is fine for some people who haven't access to html servers to put
>> some projects there, but I like also to add only the URL too. At the
>> moment I am waiting a bit how it goes on.
>>
>> Some time ago as dejanews has gone we have had a lot of trouble because
>> there was no access to the archive. If now the experience of this group
>> is splitted into rsinc forum and idl-pvwave we will lose again much of
>> the results of discussions and ideas. I don't believe that's the rsinc
>> forum is accesible by google forum search or am I wrong.
>>
>>
>>
>>> What can we do to
>>> preserve this vanishing resource? A few guidelines come to mind:
>>>

>>> 1. Use Objects. Object methods don't count against the namespace,
>>>   since they're encapsulated beneath the class name. Use unique
>>>   class names. "Box" is probably not a good name for a class.
>>>   "tvRubberBandBox" might be better.
>>
>> Later on for some new routines this is possible. But what to do with the
>> actual lib.
>>
>>
>>
>>> 2. If not using objects, imitate their namespace parsimony by
>>>   consistently prepending a class-like prefix to all related
>>>   routines...
>>>   e.g. tvRubberBandBoxDisplay.pro, or tvRBandBox_display.
>>
>> uppercases in routine names arent't accesible by unix or linux. Each of
>> these routines must be compiled first. Who has asked for this feature
>> before, if no one I will do it.
>>
>> This changing is more complicated as the changes rsinc did.
>>
>> In the past it was easy to add a new library for me because there
>> weren't much available for athmospheric science. But at the moment it is
>> difficult by source. By idl binary it isn't. This was one of the reasons
>> why I have written my compile routine. The whole plot (plotxy,plot2d ..)
>> environment of our library is stored in the plotprepare.sav file. And by
>> using the initialising script plotprepare,plot this is loaded at once.
>> After loading the sav file you have access to an info function
>> x=plotprepare_info() where you can get informations about the sources.
>> All sources are available in the catalog so you don't miss something. I
>> added to the sav file a lifetime of about I believe one month, because
>>   if you are working on a library the functionality get more
>>   improvements.
>
>
> This of course doesn't really solve the problem at all, but merely
> circumvents it in a confining way. By creating a SAV file of your code,
> you've just found a clever way to put your routines in front of other
> routines on the !PATH. The routine shadowing still exists, but you've
> guaranteed yours comes out on top.
>
> What if I have *another* plotxy command from another library (or one I
> created myself)? After I load your binary, that will no longer be
> accessible. The idea is to "preserve", not "capture" the namespace.
> Another route along these lines many packages have gone is to provide a
> shell script which sets or modifies !PATH before running IDL with the
> custom package. This provides a similar namespace "capture", but suffers
> from an even worse form of the same problem: now I can't use *any* of my

> other routines from other libraries when running the package!

I know the problem exactly and I don't found a long term solution.
The example with the legend.pro of Craig shows excactly what happens.
A tool which is needed by everyone doesn't belong to the namespace.
Someone is able to write it and later someone improves this with the
same name but different usage.

I believe it is nearly impossible to give a good every time working rule
to people which started idl and are not interested at this point into
this problem which normally later occurs.

So I am looking for another solution to incorporate additional
libraries. If I am interested in a library I did first a scan to all
files looking if I get trouble by existing names.
If only small amount of routines seems to have duplicated names I am
looking carefully in this routines to determine if they are subroutines
and often used or if it's the caller.
If it's the caller it makes no problems to change the name and to write
a comment into the routine. If not I try to separate the routines which
I really interested and so on.

This all is the same problem but I believe it should be possible
to rename all new duplicated routines locally and the calling sequence
in the files where they used.

The next problem comes by working with the new library if you did
changes in an external library routine because you have a large number
of routines written using some functions of this library.
For example in the past our complete plot environment for time axis was
based on Ray Sterners timeaxis. By working on our tools we learned how
to improve timeaxis. We did always sent a note to Ray. But he did some
more different changes to his routine and with the next library update
our plot environment works different.

What I learned by this was if we use library routines of others to build
another library it is not clear what happens by an upgrade of the
external library. I believe all of us gave their best. But sometimes a
routine is wrong named (e.g. I did the mistake to opposite name is_odd
and is_even) or two of us get's the same idea of a routine but we decide
later on to official use the new one.

Then it is difficult to upgrade an existing library.
Another reason which it makes difficult to upgrade is we do have much
more routines for private usage which are not included into the library.
I don't know what exactly someone of my colleague is using from all

libraries to calculate his measurements.
Only programs or routines which are interested for others are in the lib. Some routines of them are not regular called only after a balloon flight in Kiruna. Then the routine should exactly work as the last time.

So it's not only the problem to incorporate a library. Sometimes I wish to use a Version Control System on some server where we all develop our sources. (It's the idea of sourceforge)

Here is a tool which could help a bit

One of my case tools is able to find all dependencies of one routine depending of the search path. If the path is set only to the new library it should be easy to rename all routines which are conflicting to others in the new library and to change all calls. I am thinking about setting a general prefix for all routines.

I started thinking about this because if I add the whole package of Dominik there are five routines duplicated in the existing library. Some of them are in different versions available.

best wishes

Reimar

--
Reimar Bauer

Institut fuer Stratosphaerische Chemie (ICG-I)
Forschungszentrum Juelich
email: R.Bauer@fz-juelich.de
---------- -------
a IDL library at ForschungsZentrum Juelich
http://www.fz-juelich.de/icg/icg-i/idl_icglib/idl_lib_intro. html
========== =======

Subject: Re: Namespaces (redux)
Posted by Craig Markwardt on Wed, 25 Sep 2002 23:28:05 GMT
View Forum Message <> Reply to Message

JD Smith <jdsmith@as.arizona.edu> writes:
[...]
> Markwardt:
>> It is notable that in almost every case, the offending entity for these
>> types of namespace clashes is none other than Research Systems

>> Incorporated. We should be preaching to them.
>
> On that point I fully agree. And I have. I'm sure RSI feels a
> distinct and righteous ownership of the IDL namespace and believes
> that routines in !DIR are free to trample all over it. Here's an
> interesting random bit of output from IDLWAVE's shadow detection:
[...]

Interesting. Your shadow detection code would probably detect some problems with my library routines. For distribution I occasionally insert one routine into another. In my own library I keep them separate, but for distribution, I don't want people to have to bother with too many files, so sometimes they are merged. So you will detect DEFSUBCELL in more than one file, for example.

Craig

Subject: Re: Namespaces (redux)
Posted by Stein Vidar Hagfors H[1] on Thu, 26 Sep 2002 16:23:19 GMT
View Forum Message <> Reply to Message

Struan Gray <struan.gray@sljus.lu.se> writes:

> JD Smith, jdsmith@as.arizona.edu writes:
>
>> 1. Use Objects.
>
> I think this is the key. If I were ever to put together a library
> of routines suitable for public use (oink oink, flap, flap) the most
> natural way seems to create a package object and make all the routines
> methods. There is no law that says an object's methods have to act on
> its internal data. Converting existing libraries and the code that uses
> it could probably even be automated.

Hear, hear, all of the above. Think "toolbox/toolkit". You can have several toolboxes, and use different screwdrivers (errr.., LEGEND routines) from separate toolboxes at you own whim, mingled together in the same routine at different places. True, an essentially similar effect can be achieved with prefixes like MPFIT, but at the expense of limiting the descriptiveness and uniqueness of the prefix (versus using a class name), if one wants to avoid very long routine names. In this sense, the variable name used to hold the instantiation of a toolbox provides a means of having a localized shortcut without poisoning the namespace for other local routines.

The trouble with automated conversion is interdependencies (calls within a library must be changed to (inserting "self->" in front of those), and "user" routines must be changed, involving a choice between libraries in cases of

conflict (currently done by path manipulation or other "automagic" means), adding variables to hold each toolbox without messing up the existing routine, and finally modifying the routine calls (potentially messing up indentation/line lengths, etc). Converting the routine/file names (even in a large hierarchy of library routines) is relatively trivial, though!

Namespace crowding is one of the reasons why I every now and then hold back quite a bit from volunteering some of my own solutions to some of the problems presented in this newsgroup. For convenience and historical reasons, most of my routines involve *some* dependency on the SolarSoft library, which quite frankly is a mess in this respect! So, rather than going through the motions of ripping out those dependencies before suggesting "my" solution, I just shut up (the alternative is asking people to install a centipede to get a single pair of legs)!

--

---------- ----------
Stein Vidar Hagfors Haugan
ESA SOHO SOC/European Space Agency Science Operations Coordinator for SOHO

NASA Goddard Space Flight Center, Email: shaugan@esa.nascom.nasa.gov
Mail Code 682.3, Bld. 26, Room G-1, Tel.: 1-301-286-9028/240-354-6066
Greenbelt, Maryland 20771, USA. Fax: 1-301-286-0264
---------- ----------