

---

Subject: Namespaces (redux)

Posted by [JD Smith](#) on Tue, 24 Sep 2002 23:42:58 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

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. 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.
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.
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.

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.
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.

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

---

---

Subject: Re: Namespaces (redux)

Posted by [JD Smith](#) on Wed, 25 Sep 2002 20:19:21 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

On Wed, 25 Sep 2002 12:41:51 -0700, Reimar Bauer wrote:

> 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 dejaneux 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 aren'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 atmospheric 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 exactly 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.

This is an excellent explanation of the types of difficulties which arise from namespace pollution (to co-opt another hot-button issue). The only effective methods to combat this, in my view, all relate to changing our naming habits now. If you borrow and expand upon a piece of code from someone else, they've marked the name as their own (similar to how my dog marks the neighbor's sidewalk?), and you should respect that by changing the name, or convincing the original author to integrate your changes. Just releasing it again under the same name invites the kinds of problems you describe.

Thanks,

JD

---

---

Subject: Re: Namespaces (redux)  
Posted by [R.Bauer](#) on Thu, 26 Sep 2002 07:19:26 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

JD Smith wrote:

> On Wed, 25 Sep 2002 12:41:51 -0700, Reimar Bauer wrote:

>

>

>> 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 aren't accessible 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 atmospheric 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 exactly 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.  
>  
>  
> This is an excellent explanation of the types of difficulties which arise  
> from namespace pollution (to co-opt another hot-button issue). The only  
> effective methods to combat this, in my view, all relate to changing our  
> naming habits now. If you borrow and expand upon a piece of code from  
> someone else, they've marked the name as their own (similar to how my dog  
> marks the neighbor's sidewalk?), and you should respect that by changing  
> the name, or convincing the original author to integrate your changes.  
> Just releasing it again under the same name invites the kinds of problems  
> you describe.

I know. But also I see no chance at the moment to rename all of our sources. At the moment we did only this changes if a name conflict to a new rsinc routine is given.

By getting file\_search last time by idl 5.5 I have nearly 100 routines in the library altered and tested.

As you know my legend routine is named rb\_legend so in some cases I have already named routines as you proposed.

Another idea for this point I have is I don't know if it is possible but I wish to have something like the autocompletion.

For example if I got the idea to name a routine `legend@rb.pro` and I have no different routine in the library which uses the same keys (`legend*`). Then it should be possible to call it with `legend`. And if there are others I should be informed and I have to choose between.

I like to have the same functionality for routines as for keywords. If this rule is possible it would be very easy to rename all files of an existing library for us too.

Reimar

In the next IDL training course end of oktober I will give this informations to the new idl programmers.

.

>  
> Thanks,  
>  
> 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](http://www.fz-juelich.de/icg/icg-i/idl_icglib/idl_lib_intro.html)  
=====

---

Subject: Re: Namespaces (redux)  
Posted by [JD Smith](#) on Thu, 26 Sep 2002 14:58:29 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

On Wed, 25 Sep 2002 16:28:05 -0700, Craig Markwardt wrote:

> 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.

That's right... all the dirty little secrets and long forgotten skeletons  
come tumbling out of the closet under the eagle-eyed scrutiny of IDLWAVE.  
Actually, the other day I found that I had three modified copies of  
XMANAGER lying about on my path (don't ask), and very occasionally one of  
these butchered alternatives was getting called. So even if the world of  
IDL users suddenly repented and created an international naming tribunal  
to adjudicate and settle conflicts, shadow detection would still have a  
place ;)

JD

---

Subject: Re: Namespaces (redux)  
Posted by [William Daffer](#) on Fri, 27 Sep 2002 22:50:51 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

I meant to reply to this in the newsgroup, but mistakenly sent it by email to J.D. Smith. (sorry)

What I said in that email is:

Personally, I'd love to see RSI spend some time and effort on the language itself and implement some import semantics. I think that would go a long way to solving this problem.

But I doubt they will, it isn't \*glitzy\* enough.

whd

--

MEDICINE, n. A stone flung down the Bowery to kill a dog in Broadway.

-- Ambrose Bierce: \_The Devil's Dictionary\_

---