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 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.

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
