Subject: Nice ways to compile
Posted by Robert Barnett on Sun, 10 Apr 2005 02:38:37 GMT
View Forum Message <> Reply to Message

Hi,

I'm looking for an easier way to compile my code into a .sav file. It would save a considerable of time an effort if compilation was just one step only. It would also make it much more repoducible and less bug prone.

I've included a synopsis of how I go about things starting with my development workflow:

* Program concept and design
* Demonstrate concept using snippets of code and interactive mode
* Package the code so it can be easily accessed and reused
* Test the package, particularily for interopability with other pakcages.
* Continually refine test and refine package until it does what it is supposed to do
+ Complile the package and depenancies to a save file and run in an IDL VM "sandbox" environment.
* Briefly test that it works in the VM as I would expect it to.
* Install application on target computer.

I develop using emacs. I expect to be able to compile my IDL packages with a single UNIX script (step marked as a +). I also expect that my compilation depends solely on the contents of the filesystem rather than the current state of the IDL command prompt. This ensures that I can always return to a snapshot of my idl directory as it was when I compiled my code.

For example, I currently compile like the following:

./compiler.pl ../deployed/ra_dualrenal.sav robbies_tools/ snaptools/ renal_analysis/

compiler.pl is just a simple perl script with the syntax:

./compiler <save file> <path1> <path2> ...

compiler.pl compiles all .pro files in any sub-directories in the given paths. I always run a cvs commit or do a backup of my IDL directory when I do this step.

I appreciate that
there are smart compilation scripts out there, but I prefer to specify

each path individually because I use call_method, call_procedure, call_function and obj_new quite a bit. I also cannot afford to find out that I missed a dependacy buried somewhere in a widget control. This is particularly troublesome after I've installed the IDL runtime application at a hospital which is an hours drive away.

Specifically, I've had problems reliably compiling some IDL graphics classes for use on the IDL VM. I thought that running IDLITRESOLVEITOOLS might do the trick, but it evidently hasn't. At the same time I don't really want to compile the entire IDL library every time. I just want to resolve the IDL graphics classes which I want to use (or at least just compile a single category of classes).

Perhaps I am looking at this problem completely wrong. I would expect there to be some way I can append a compiler directive which indicates that blah__define.pro requires 'idlgrlegend__define'. Compiler directives are not really in the IDL idiom, so I'm wondering what other choices I might have.

Cheers,
Robbie

--

nrb@ Robbie Barnett
imag Research Assistant
wsahs Nuclear Medicine & Ultrasound
nsw Westmead Hospital
gov Sydney Australia
au +61 2 9845 7223