
Subject: Re: Project newbie asks...well what about ENVI projects?

Posted by [jeffnetles4870](#) on Fri, 25 Jul 2008 15:50:43 GMT

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

On Jul 25, 10:50 am, David Fanning <n...@dfanning.com> wrote:

> Jeff N. writes:

>> Now, however, i find myself writing a bunch of code that i want to
>> distribute to several people, and the idea of setting up a project
>> immediately starts to seem attractive. Here's the thing though....the
>> code i'm writing is for ENVI. Basically i've got a bunch of .pro
>> files that adds a new menu item to ENVI's main menu and then handles
>> all the commands on that new menu item. So I think the relevant
>> difference here between what i want to do and a normal (?) IDL project
>> is that I don't want all of my .pro files to end up compiled into one
>> big save file that you launch with a single command. I want to have
>> one save file created for each .pro file. Is that possible? Is it
>> possible to have the directory that the compiled code ends up in be
>> different than the directory where the .pro files are (but maintain
>> the directory structure)?

>

>> I also have a resources folder that contains text files, images,
>> spectral libraries, etc.....stuff that isn't code but still needs to
>> end up in the "output" directory where all the compiled code ends up.

>

>> In the alternative, should i be thinking about how i could write my
>> own "build" program, that searches for .pro files, calls SAVE for
>> them, etc. and then handles the resource directory?

>

>> Any suggestions would be great.

>

> This was a topic of conversation at yesterday's Extending
> ENVI seminar. No one appears to be sure about the right
> way to do this. One of the most sensible suggestions I
> heard was to add your ENVI save-add folder to your
> Workspace as a project, but to make sure it is NOT
> on your IDL path.

>

> This way, you can see the contents, drag your completed
> save and pro files into it, etc., but not have to deal
> with it otherwise.

>

> I think you are going to have to hand-save the sav files
> you want to add to ENVI. I would just do this in the normal
> way in whatever project folder you are using at the moment,
> then drag these files over to the save-add folder.

>

> Note that there is a new way to create the ENVI buttons
> now to make your added programs accessible to ENVI. You

```

> no longer have to modify the ENVI menu text file.
>
> Your user function has to have two modules in it: the
> normal event handler module that gets called when the
> action button is selected, and a procedure defined with
> the same name as the event handler with a "_define_buttons"
> attached to it:
>
>   PRO MyFunction_Define_Buttons, buttonInfo
>   .....
>   END
>
>   PRO MyFunction, event
>   .....
>   END
>
> Inside the "define buttons" procedure, you use the ENVI
> routine ENVI_DEFINE_MENU_BUTTON to create your buttons.
> Keywords allow you to name and position the button in
> relation to other buttons, etc.
>
> Apparently, this has been an undocumented feature for a
> long time, but it has been exposed in this release. It is
> very slick, I think.
>
> Cheers,
>
> David
> --
> David Fanning, Ph.D.
> Fanning Software Consulting, Inc.
> Coyote's Guide to IDL Programming: http://www.dfanning.com/
> Sepore ma de ni thui. ("Perhaps thou speakest truth.")

```

Thanks David.

I've definitely used the `_Define_Buttons` routines. For my particular application, however, i've done things a little differently than the standard method of putting the "define buttons" procedure at the top of the .pro file containing the procedure you're trying to add a menu item for. I created a separate pro file that does nothing but call the define-buttons ENVI command, and has a little placeholder procedure at the bottom.

So my "Menu.pro" file has this in it:

```

PRO MyApp_Menu_Define_Buttons, buttonInfo
;do all the menu items for the entire app
END

```

```
PRO Myapp_Menu  
  ;do nothing at all  
END
```

The reason I did this is because 1) it just seemed like an easier way to make sure that the top-level part of my menu always gets created and the lower-down menu items know that the top level menu item is there. and 2) Some menu items are generated or not-generated using conditional logic. I have some spectral library stuff distributed in text files, so i have logic that only creates the menu to process the spectral libraries if the spectral libraries are actually present. 3) It also lets me put debug commands in the menu that only i see, but normal users don't see when all the code gets distributed. Definitely handy stuff!

Jeff
