
Subject: Including files/eps into PS

Posted by [DMottershead](#) on Wed, 15 Apr 1998 07:00:00 GMT

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

Is there a way of creating an EPS/CGM file (or any other file type), storing and later opening and including this file in a PS output. What I am trying to do is plot some data and store this plot for later. When needed I would like to open the plot and place a border around it. I would also like to be able to open the plot again and view it say on the screen. Would it be as simple as opening the EPS file and appending the border onto the end of the file? I tried this but it didn't seem to work so I don't think this would be the way to go.

-----== Posted via Deja News, The Leader in Internet Discussion ==-----

<http://www.dejanews.com/> Now offering spam-free web-based newsreading

Subject: Re: Including files

Posted by [J.D. Smith](#) on Mon, 24 May 1999 07:00:00 GMT

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

Nils Johnson wrote:

>
> The IDL documentation mentions include files, which allows "statements
> contained in an include file" to be "textually inserted into the including
> file."
>
> However, I can not find any mention or examples of how, exactly, to
> include other files. What is the syntax? Are there any examples in the
> documentation or sample code that comes with IDL?
>
> Nils

@foo

includes the file foo (or foo.pro) which exists in the search path. The usual search order issues apply (multiple files foo lead to unexpected results!).

One way this can be used to advantage is with common block definitions. If there is not a known entry point for defining a common block which is used by various different procedures, you must either define it beforehand in some startup routine which the user must remember to run or put in their startup file, or else simply define the common block in all routines in which it's used. That is, you must use:

common cblock, var1, var2, var3, var4, ...

everywhere instead of:

common cblock

Normally this is a real pain, since changing the common block requires changing all of its definition statements. However, if you put the definition in one file and simply do a:

@cblock

at the beginning of all relevant procedures, you can change your common block in one place and still not require an initializer. Of course, eliminating the common block is another good solution, but sometimes this can't be done.

This technique can be extended to other areas also, such as common error checking code, etc.

Good Luck,

JD

--

J.D. Smith |*| WORK: (607) 255-5842
Cornell University Dept. of Astronomy |*| (607) 255-6263
304 Space Sciences Bldg. |*| FAX: (607) 255-5875
Ithaca, NY 14853 |*|

Subject: Re: Including files

Posted by [Craig Markwardt](#) on Mon, 24 May 1999 07:00:00 GMT

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

nilsj@unixg.ubc.ca (Nils Johnson) writes:

>
> The IDL documentation mentions include files, which allows "statements
> contained in an include file" to be "textually inserted into the including
> file."
>
> However, I can not find any mention or examples of how, exactly, to
> include other files. What is the syntax? Are there any examples in the
> documentation or sample code that comes with IDL?

It is not particularly good programming style to do this, and hence I
use it all the time! :-)

Here is you you do it:

```
pro mypro, arg1, arg2
```

```
    arg1 = arg2 + 5  
    ... etc  
@myinclude.pro
```

```
    return  
end
```

Note that the standard path is consulted in including myinclude.pro, otherwise it should be in your current directory. Text is included from the file verbatim. In the example above, @myinclude.pro appears within a procedure. Therefore, myinclude.pro should contain neither a PRO or FUNCTION declaration, nor a final END statement. Works the same as a C #include directive.

Happy programming,
Craig

--

Craig B. Markwardt, Ph.D. EMAIL: craigmnet@astro.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response

Subject: Re: Including files
Posted by [Liam Gumley](#) on Tue, 25 May 1999 07:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

Martin Schultz wrote:

```
> Liam Gumley wrote:  
>> So do I, but I always make sure I've had my coffee first :-)  
>>  
>> The script above generates a syntax error at the ENDIF line when  
>> included with  
>>  
>> @myinclude.pro  
>>  
> oh je! you ask for tested software? I'll sure get my coffee next time  
> round...  
> But it does work in procedures (or functions), just not in $MAIN$ or if  
> included at the prompt.  
> So at least we now know how to prevent someone from using common blocks  
> in $MAIN$ ;-)
```

Martin,

Please accept my most humble apologies; you are correct. I was mistakenly basing my comments on the IDL 5.1 and 5.2 documentation, which is apparently dead wrong (or at least incomplete). To see what I mean, check out the online book 'Using IDL', section 'Running IDL', 'Batch Execution'.

Cheers,
Liam.

--

Liam E. Gumley
Space Science and Engineering Center, UW-Madison
<http://cimss.ssec.wisc.edu/~gumley>

Subject: Re: Including files
Posted by [R.Bauer](#) on Tue, 25 May 1999 07:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

Martin Schultz wrote:

> Liam Gumley wrote:
>>
>>
>> So do I, but I always make sure I've had my coffee first :-)
>>
>> The script above generates a syntax error at the ENDIF line when
>> included with
>>
>> @myinclude.pro
>>
> oh je! you ask for tested software? I'll sure get my coffee next time
> round...
> But it does work in procedures (or functions), just not in \$MAIN\$ or if
> included at the prompt.
> So at least we now know how to prevent someone from using common blocks
> in \$MAIN\$;-)
>
> Cheers,
> Martin.
>

I will remind that's the extension .pro is not necessary but you should write a space and a comment statement behind the include statement.
If you don't it will never found on unix sytems if it's written on a NT IDL.

@myinclude ;

Cia,
R

Subject: Re: Including files

Posted by [Martin Schultz](#) on Tue, 25 May 1999 07:00:00 GMT

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

Liam Gumley wrote:

>
>
> So do I, but I always make sure I've had my coffee first :-)
>
> The script above generates a syntax error at the ENDIF line when
> included with
>
> @myinclude.pro
>

oh je! you ask for tested software? I'll sure get my coffee next time round...

But it does work in procedures (or functions), just not in \$MAIN\$ or if included at the prompt.

So at least we now know how to prevent someone from using common blocks in \$MAIN\$;-)

Cheers,
Martin.

--

|||||||\\-----// |||||
Martin Schultz, DEAS, Harvard University, 29 Oxford St., Pierce 109,
Cambridge, MA 02138 phone (617) 496 8318 fax (617) 495 4551
e-mail mgs@io.harvard.edu web <http://www-as/people/staff/mgs/>

Subject: Re: Including files

Posted by [Liam Gumley](#) on Tue, 25 May 1999 07:00:00 GMT

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

Martin Schultz wrote:

> Of course ;-) I can also offer you a solution which is to include a
> default definition in your include file as well, such as:
> -----
> ; myinclude.pro === note, this is not a procedure
> common bla,var1,var2,svar
>

```
> if (n_elements(var1) eq 0) then begin
>   var1 = fltarr(100)
>   var2 = 100L
>   svar = "
> endif
> -----
> I really like participating in this newsgroup to get me started in the
> morning!
```

So do I, but I always make sure I've had my coffee first :-)

The script above generates a syntax error at the ENDIF line when included with

@myinclude.pro

because block termination statements (ENDFOR, ENDIF, ENDREP, ENDWHILE) are only allowed in procedures or functions. So you'll need to fake it as follows:

```
common bla,var1,var2,svar
if n_elements(var1) eq 0 then $
  var1=fltarr(100) &
  var2=100L &
  svar="
```

Cheers,
Liam.

--

Liam E. Gumley
Space Science and Engineering Center, UW-Madison
<http://cimss.ssec.wisc.edu/~gumley>

Subject: Re: Including files
Posted by [Martin Schultz](#) on Tue, 25 May 1999 07:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

J.D. Smith wrote:

```
>
> [...]
>
> That is, you must use:
>
> common cblock, var1, var2, var3, var4, ...
>
> everywhere instead of:
```

>
 > common cblock
 >
 > Normally this is a real pain, since changing the common block requires
 > changing all of its definition statements. However, if you put the
 > definition in one file and simply do a:
 >
 > @cblock
 >
 > at the beginning of all relevant procedures, you can change your common
 > block in one place and still not require an initializer.

This is not quite right. You will have to make sure that your common block gets initialized correctly! For IDL, a common block at the beginning is nothing else than a list of variable names. The first procedure that uses these variables determines their type and dimensions. Once these are set, you have to exit your IDL session and restart if you want to make any changes. Unless you are certain that it is always the same procedure that makes first use of the common block variables, you are in great danger of trapping into conflicting variable declarations (implicit definitions that is).

Of course ;-) I can also offer you a solution which is to include a default definition in your include file as well, such as:

```
-----
; myinclude.pro === note, this is not a procedure
common bla,var1,var2,svar

if (n_elements(var1) eq 0) then begin
  var1 = fltarr(100)
  var2 = 100L
  svar = "
endif
-----
```

Then, regardless which procedure uses your include (myinclude that is ;-), you will always have the same type and dimension for each variable.

> Of course,
 > eliminating the common block is another good solution,
 > a much better one, I should say!

> but sometimes
 > this can't be done.
 also true. ;-)

>
> This technique can be extended to other areas also, such as common error
> checking code, etc.
>
> Good Luck,
>
> JD
>
> --
> J.D. Smith |*| WORK: (607) 255-5842
> Cornell University Dept. of Astronomy |*| (607) 255-6263
> 304 Space Sciences Bldg. |*| FAX: (607) 255-5875
> Ithaca, NY 14853 |*|

I really like participating in this newsgroup to get me started in the morning!
Regards,
Martin.

--

|||||||\\-----//////////////// //|||||||
Martin Schultz, DEAS, Harvard University, 29 Oxford St., Pierce 109,
Cambridge, MA 02138 phone (617) 496 8318 fax (617) 495 4551
e-mail mgs@io.harvard.edu web <http://www-as/people/staff/mgs/>
