Subject: Re: path to .sav file
Posted by Allan Whiteford on Mon, 23 Oct 2006 10:55:43 GMT
View Forum Message <> Reply to Message

greg michael wrote:
> I'm sure there must be any easy answer to this, but I can't figure it
> out...
>
> If a VM program is run with something like:
>
> idl -vm=...path.../myprog.sav
>
> how can I get that path from within the program?
>
> Greg
>

Greg,

It's not pretty but:

```
help,/source,output=a
idx=where(stregex(a,'myprog +([/,a-z,0-9,_,\.]*)',/fold_case) eq 0)
file=(stregex(a[idx],'myprog +([/,a-z,0-9,_,\.]*)',$
                /fold_case,/subexpr,/extract))[1]
```

I'm sure there is a better way.

Thanks,

Allan

Subject: Re: path to .sav file
Posted by greg michael on Mon, 23 Oct 2006 12:30:06 GMT
View Forum Message <> Reply to Message

Thanks very much Allan! That's just what I need. I didn't get the stregex part to work (wish I could, but it's a magic I can't fathom), but grabbing it out of the help output is fine. Pretty enough for my purposes.

```
main='myprog'
help,/source,output=a
sav_file=strtrim(strmid(a[where(strmid(a,0,strlen(main)+1) eq main+'
')],strlen(main)),2)
```

(I match an extra space to avoid a routine called 'myprogGUI')

regards,
Greg

Subject: Re: path to .sav file
Posted by btt on Mon, 23 Oct 2006 12:39:16 GMT
View Forum Message <> Reply to Message

greg michael wrote:
> I'm sure there must be any easy answer to this, but I can't figure it
> out...
>
> If a VM program is run with something like:
>
> idl -vm=...path.../myprog.sav
>
> how can I get that path from within the program?
>
> Greg
>
Hello,

I think you want somethong like Jim Pendleton's SOURCEPATH.pro which you
can find at http://www.ittvis.com/codebank

Ben

Subject: Re: path to .sav file
Posted by David Fanning on Mon, 23 Oct 2006 12:50:05 GMT
View Forum Message <> Reply to Message

Ben Tupper writes:

> I think you want somethong like Jim Pendleton's SOURCEPATH.pro which you
> can find at http://www.ittvis.com/codebank

Or, PROGRAMROOTDIR, here:

   http://www.dfanning.com/programs/programrootdir.pro

It's based on Jim's, but with some tweaks I need for
the way I distribute programs.

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.")

Subject: Re: path to .sav file
Posted by greg michael on Mon, 23 Oct 2006 13:06:19 GMT
View Forum Message <> Reply to Message

Now I have many to choose from!

Seems Jim's has been updated to use Scope_Traceback(), which probably
makes it more robust than the 'help'-based methods.

many thanks,
Greg

Subject: Re: path to .sav file
Posted by David Fanning on Mon, 23 Oct 2006 13:41:03 GMT
View Forum Message <> Reply to Message

greg michael writes:

> Seems Jim's has been updated to use Scope_Traceback(), which probably
> makes it more robust than the 'help'-based methods.

Certainly makes it a lot simpler to write the code!
I do notice, though, that the path that comes back
doesn't have a file separator on the end of the path.
You will have to remember to append this is you are
going to use the path to locate another file, etc.

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.")

Subject: Re: path to .sav file
Posted by btt on Mon, 23 Oct 2006 13:57:54 GMT
View Forum Message <> Reply to Message

David Fanning wrote:
> greg michael writes:
>
>> Seems Jim's has been updated to use Scope_Traceback(), which probably
>> makes it more robust than the 'help'-based methods.
>
> Certainly makes it a lot simpler to write the code!
> I do notice, though, that the path that comes back
> doesn't have a file separator on the end of the path.
> You will have to remember to append this is you are
> going to use the path to locate another file, etc.
>

Hi,

Jim added the _EXTRA keyword which is passed to FILE_DIRNAME so you can
simply pass the keyword /MARK_DIRECTORY in to have it done for you.

Ben

Subject: Re: path to .sav file
Posted by Douglas G. Dirks on Mon, 23 Oct 2006 16:06:58 GMT
View Forum Message <> Reply to Message

Ben Tupper wrote:
> greg michael wrote:
>> I'm sure there must be any easy answer to this, but I can't figure it
>> out...
>>
>> If a VM program is run with something like:
>>
>> idl -vm=...path.../myprog.sav
>>
>> how can I get that path from within the program?
>>
>> Greg
>>
> Hello,
>
> I think you want somethong like Jim Pendleton's SOURCEPATH.pro which you
> can find at http://www.ittvis.com/codebank
>
> Ben

ROUTINE_INFO will give you the path to the source of any compiled
routine, provided you know its name, and it works all the way back
to IDL 5.0:

```
routinepath = ROUTINE_INFO('myprog', /SOURCE)
```

(assuming that myprog.sav has a main routine called 'myprog'...)

Add in FILE_DIRNAME (IDL 6.0 or later) and you've got what you want,
I think:

```
routinedir = FILE_DIRNAME(routinepath.path)
```

You can use the MARK_DIRECTORY keyword to FILE_DIRNAME to get a
trailing directory separator.

Nothing against SOURCEPATH or PROGRAMROOTDIR, but if all you want
is to know the path from which a routine was compiled, you can do
it with IDL internal routines.

Doug

Subject: Re: path to .sav file
Posted by David Fanning on Mon, 23 Oct 2006 16:22:08 GMT
View Forum Message <> Reply to Message

Douglas G. Dirks writes:

> Nothing against SOURCEPATH or PROGRAMROOTDIR, but if all you want
> is to know the path from which a routine was compiled, you can do
> it with IDL internal routines.

Indeed. As either of those two programs would prove to you. :-)

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.")

Subject: Re: path to .sav file

Posted by David Fanning on Mon, 23 Oct 2006 16:27:46 GMT

View Forum Message <> Reply to Message

David Fanning writes:

> Indeed. As either of those two programs would prove to you. :-)

Some people just insist on using one well-named program
when 8-10 lines of code would do just fine. As Coyote
says, "It's a damn commie plot!"

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.")

Subject: Re: path to .sav file
Posted by JD Smith on Mon, 23 Oct 2006 20:22:18 GMT

View Forum Message <> Reply to Message

On Mon, 23 Oct 2006 10:27:46 -0600, David Fanning wrote:

> David Fanning writes:
>
>> Indeed. As either of those two programs would prove to you. :-)
>
> Some people just insist on using one well-named program
> when 8-10 lines of code would do just fine. As Coyote
> says, "It's a damn commie plot!"

Looks like 1 line to me ;). I use a modification of this method: create a
little batch file (e.g. project_directory.pro) with something like:

```
common my_project_dir, proj_dir
if n_elements(proj_dir) eq 0 then begin
   resolve_routine,'myproj',/NO_RECOMPILE
   proj_dir=(routine_info('myproj',/SOURCE)).PATH
   ps=path_sep()
   if strmid(proj_dir, 0,1) ne ps then $ ;relative filename
      proj_dir=file_expand_path(proj_dir)
endif
```

Then in any routine where you need to know the project directory, just

@project_dir

at the top. More real world examples would locate the interesting
relative paths within the distribution. You can use anything for
'myproj', just pick a routine which is in a known useful location. This
works for source, compiled SAV's, anything. For maximum usefulness, have
the relative location of the routine (in source distributions) and the
.sav file (in binary distributions) the same.

JD

Subject: Re: path to .sav file
Posted by David Fanning on Mon, 23 Oct 2006 21:21:11 GMT
View Forum Message <> Reply to Message

JD Smith writes:

```
>  I use a modification of this method: create a
>  little batch file (e.g. project_directory.pro) with something like:
>
>  common my_project_dir, proj_dir
>  if n_elements(proj_dir) eq 0 then begin
>     resolve_routine,'myproj',/NO_RECOMPILE
>     proj_dir=(routine_info('myproj',/SOURCE)).PATH
>     ps=path_sep()
>     if strmid(proj_dir, 0,1) ne ps then $ ;relative filename
>        proj_dir=file_expand_path(proj_dir)
>  endif
>
>  Then in any routine where you need to know the project directory, just
>
>  @project_dir
>
>  at the top.  More real world examples would locate the interesting
>  relative paths within the distribution.  You can use anything for
>  'myproj', just pick a routine which is in a known useful location.  This
>  works for source, compiled SAV's, anything.  For maximum usefulness, have
>  the relative location of the routine (in source distributions) and the
>  .sav file (in binary distributions) the same.
```

I've tried distributing files that I @filename
into the code, but this always causes me grief
when people try to run the code because it requires
that the directory where the @ files are located
be on the PATH (or in the local directory).

For some reason, people with UNIX machines have
a *great* many problems getting their paths set
up properly, in my experience. More often than not,
my files are in 5 different directories on their
machines, and I can easily spend an entire day
getting things on their path sorted out. :-(

> Looks like 1 line to me ;)

I think it was two. But I try not to let the facts
get in the way of making a point, whenever possible. :-)

(For some reason, I was thinking of TV and TVIMAGE
when I wrote that response. It's true that TVIMAGE
uses ONLY built-in IDL commands.)

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.")