
Subject: Re: File_ Which What!?
Posted by [JD Smith](#) on Mon, 25 Nov 2002 22:00:27 GMT
[View Forum Message](#) <> [Reply to Message](#)

On Fri, 22 Nov 2002 07:41:22 -0700, Jaco van Gorkom wrote:

```
> "David Fanning" <david@dfanning.com> wrote in message
> news:MPG.184759add416747d989a42@news.frii.com...
>> Mark Hadfield (m.hadfield@niwa.co.nz) writes:
>>
>>> This would be sensible, *if* FILE_WHICH was to be used only to search
>>> for .pro or .sav files. There is nothing in the documentation that
>>> says this is the case, and I (and David) have used it to search for
>>> files with other extensions. I'm just glad it was David who got
>>> tripped up by the hidden gotcha. (Not that I'm not sympathetic,
>>> David, but...)
>>
>> Well, maybe there should be an EVERYDARNONE keyword, to force it to
>> search everything. Now that I think about it, why would you have files
>> on your path that you didn't want to search? Maybe I'll revisit this
>> with my source. :-)
>
> Why are you using FILE_WHICH to *search* for files in the first place?
> FILE_SEARCH (or FINDFILE) would seem a better candidate. Anyway,
> FILE_WHICH should tell you exactly *which* file IDL would find, if it
> should ever go looking for a fire.clb file on its path. Directories
> without .pro or .sav files (at IDL startup time!!) simply are not
> included in the IDL path, states the Help for !PATH.
>
> sympathetically yours,
> Jaco
>
> PS: As I started using IDL, I frequently ran into problems that I could
> only solve by exiting and restarting IDL. Later I discovered many of the
> reasons, and this was one of them.
```

FILE_WHICH simply scrubs through a list of paths and looks for the first existing file specified in one of them. This is fine if you know which file you're looking for. And FINDFILE (with FILEPATH) lets you find any number of files by pattern (like '*.clb'), assuming you know the directory(ies) they are in.

But what to do if you don't know where they are, but you need all '*.clb' files from some special directory? My solution has been to place alongside the target files a special marker routine (e.g. `irs_calib_dir_marker.pro`). This dummy routine is compiled, and ROUTINE_INFO is used to get its source, from which the path is derived and placed in a common block. Ugly, but pretty failsafe. It also

ensures you're getting exactly the files you meant to. I bundle it all up into an include file, which I "@" at the top of any routine which needs the directory. It looks like:

```
common irs_dir, irs_calib_dir
if n_elements(irs_calib_dir) eq 0 then begin
    irs_calib_dir_marker
    source=(routine_info('irs_calib_dir_marker',/SOURCE)).PATH
    dir=strmid(source,0,strpos(source,path_sep(),/REVERSE_SEARCH))
    if NOT file_test(dir,/DIRECTORY) then $
        message,'Cannot locate IRS calibration data directory: '+dir
    irs_calib_dir=dir
endif
```
