Subject: Re: FINDFILE and Unix
Posted by tam on Wed, 02 Feb 2000 08:00:00 GMT
View Forum Message <> Reply to Message

Findfile on unix is rather more of a direct invocation of ls than
you might imagine.

E.g., to get only files and directories in a given directory without
looking in subdirectories...

files = findfile('-d somedirectory/*')

I.e., you can specify the -d qualifier for the ls command -- and
as far as I know any other ls qualifier, e.g., you can use -l to get
the long directory listing...

Of course I am using an archaic version of IDL 5.0 on a DEC alpha
and this may have been fixed.

Another couple of gotchas... You may run into the Unix limit on the
length of a command when searching directories with many files.
Also, when searching multiple subdirectories you can find nasty
little directory headers in your file list (as well as blank separators).

I do hope these have been fixed in later versions of IDL but you
may want to check.

Regards,
Tom McGlynn

Doug Reynolds wrote:
>
> I recently had some problems with the FINDFILE function. First of all, the
> Online Help documentation contains an error:
>
> Under UNIX, to refer to all of the files in a directory only, use
> FINDFILE('/File_Specification/*.*'). To include all the files in any
> subdirectories, use FINDFILE('/File_Specification/*')
>
> However, under Unix, a call with "*.*" omits any files that do not include "."
> in their name. In addition, searching for "*.*" does not prevent searching
> through subdirectories - if a directory name contains ".", FINDFILE will
> include any files within it.
>
> Now, the reason I was looking through the documentation in the first place is
> that FINDFILE can not do what I was trying to do. We have a directory that
> contains 97 subdirectories starting with "f" - "f300", "f301", "f302", etc. I
> wanted to get a list of all of these subdirectories. Here is what happened:

>
> IDL> files = findfile ("/users/username/f*")
> IDL> help, files
> FILES STRING = Array[1767]
>
> The problem is that in addition to returning the directory names I wanted,
> FILES also included all the files and subdirectories in these directories.
> Unfortunately, under Unix, there is no way to make FINDFILE return a directory
> name without also including the contents.
>
> My guess is that FINDFILE is just an interface to the "ls" command. If so, I
> think it would be nice if a keyword could be added to make FINDFILE execute
> either "ls" or "ls -d", depending on whether or not the user wants to include
> subdirectory contents. For example:
>
> files = findfile ("/users/airi/f*", /norecurse)
> files = findfile ("/users/airi/f*", /recurse)
>
> In the meantime, I have written a FINDFILE replacement, which I have included
> below. On an OS other than Unix it just calls FINDFILE, but Unix users have
> the option of using a /RECURSE keyword to enable or disable subdirectory
> searches. Any comments / feedback would be appreciated.
>
> Doug Reynolds
>
> ;+
> ; NAME:
> ; llfindfile
> ; PURPOSE:
> ; Replacement for IDL's FINDFILE function, which does not work properly
> ; under Unix.
> ; EXAMPLES:
> ; 1. List all files ending with .dat in the current directory:
> ; files = llfindfile ('*.dat')
> ; 2. List all files in the entire /users/airi hierarchy, and return
> ; the number of entries:
> ; files = llfindfile ('/users/airi', /recurse, count = count)
> ; CALLING SEQUENCE:
> ; files = llfindfile (filespec, [/recurse] [,count=entries])
> ; INPUTS:
> ; path The file specification to match (can include wildcards)
> ; OPTIONAL INPUTS:
> ; KEYWORD PARAMETERS:
> ; /help Prints this header
> ; /recurse If set, causes the search to include files and
> ; subdirectories within matched directories
> ; OUTPUTS:
> ; OPTIONAL OUTPUTS:

```
>  ;      count        Returns the number of files found
>  ; COMMON BLOCKS:
>  ; SIDE EFFECTS:
>  ;      IDL's FINDFILE appends a colon to a name if it is a directory; this
>  ;      routine appends a '/'.
>  ; RESTRICTIONS:
>  ; PROCEDURE:
>  ; MODIFICATION HISTORY:
>  ;      000131  DSR  Written.
>  ;-
>  function llfindfile, path, recurse = recurse, count = count, help = help
>
>  ;   Help
>
>     if keyword_set(help) then begin
>        doc_library, 'llfindfile'
>        return, 0
>     endif
>
>     if !version.os_family ne 'unix' then begin
>        return, findfile (path, count = count)
>     endif else begin
>        if n_elements (path) eq 0 then command = 'ls -F' else begin
>           if not keyword_set (recurse) then command = 'ls -Fd ' else $
>             command = 'ls -F '
>           command = command + path
>        endelse
>        spawn, command, result
>        count = n_elements (result)
>        return, result
>     endelse
>  end
```