Subject: Re: find_subtree.pro
Posted by davidf on Wed, 03 Jun 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Kristian Kjaer (kristian.kjaer@risoe.dk) writes:

> I am looking for piece of code - similar to findfile() - which will
> return the fully-qualified paths to all subdirectories of the default
> directory, or return the fully-qualified paths to, say, all files *.dat
> in all subdirectories of the default directory.
>
> (I use IDL 5 on WinNT.)

Here is a little thing I coded up today when I should have
been doing something a whole lot more useful. I still don't
fully understand recursive functions, so these kinds of things
always suck me in. :-(

Anyway, this appears to work under the extensive testing
I've subjected it to. :^)

It is specific for Windows machines. The names that it
returns are given relative to the target directory name
(to which I apply NO error checking!). It will, perhaps,
give you some ideas.

As always, I'm open for gentle criticism, which I will
probably deserve for this one. :-)

Cheers,

David

```
 ********************************************************** *****
Function AllDir, target

; This function returns the names of all the directories
; rooted at the "target" directory. The function is specific
; for the Windows operating system. The return names are
; given with respect to the target name.

    ; Default directory is current directory.

IF N_Params() EQ 0 THEN BEGIN
   CD, Current=target
   target = target + '\'
ENDIF
```

```
    ; Switch to target directory.

CD, target, Current=thisDirectory

    ; Find the files in the target directory.

theseFiles = Findfile('*', Count=count)
IF count EQ 0 THEN RETURN, ''

    ; Find the directories in the file list. Directories
    ; end with a "\" character.

endCharPos = StrLen(theseFiles) - 1
FOR j=0,count-1 DO BEGIN
   IF theseFiles[j] NE '.\' AND theseFiles[j] NE '..\' THEN BEGIN
      lastChar = StrMid(theseFiles[j], endCharPos[j], 1)
      IF lastChar EQ '\' THEN BEGIN
         IF N_Elements(theseDirs) EQ 0 THEN $
            theseDirs = [theseFiles[j], AllDir(theseFiles[j])] ELSE $
            theseDirs = [theseDirs, theseFiles[j], AllDir(theseFiles[j])]
      ENDIF
   ENDIF ELSE theseDirs = ''
ENDFOR

   ; Add the target name.

theseDirs = target + theseDirs

   ; Go back to the starting directory.

CD, thisDirectory

    ; Remove null strings and non-unique values.

returnValue = theseDirs[Where(theseDirs NE target) > 0]
returnValue = returnValue[Uniq(returnValue)]

    ; Return the list.

RETURN, returnValue
END
```

David Fanning, Ph.D.
Fanning Software Consulting
E-Mail: davidf@dfanning.com
Phone: 970-221-0438
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: find_subtree.pro
Posted by mallors on Wed, 03 Jun 1998 07:00:00 GMT
View Forum Message <> Reply to Message

In article <35741201.F0CD2D26@risoe.dk>,
 Kristian Kjaer <kristian.kjaer@risoe.dk> writes:
> Hi,
>
> I am looking for piece of code - similar to findfile() - which will
> return the fully-qualified paths to all subdirectories of the default
> directory, or return the fully-qualified paths to, say, all files *.dat
> in all subdirectories of the default directory.
>
> (I use IDL 5 on WinNT.)
>

Here is a program I call FINDFILES that I wrote, since IDL's
FINDFILE seems somewhat limited. Unfortunately, I only had
access to Unix and VMS machines, so for Windows and Mac
the program is currently just a wrapper to FINDFILE. Perhaps
someone on one of those machines can update it?

-bob

```
; start FINDFILES.PRO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;+
; NAME:
;    FINDFILES
;
; PURPOSE:
;    Find all files matching a file filter.  Replacement for the
;    IDL builtin routine FILEFILE, which does not handle recursive
;    search of directories correctly.
;
;    Currently implemented for UNIX and VMS systems only.  For Windows
;    and MacOS, this routine is a wrapper for FINDFILE.
;
; TYPE:
;    FUNCTION
;
; CATEGORY:
;    FILES
;
; CALLING SEQUENCE:
;    result = FINDFILES (fileFilter [, /RECURSE, ROOT = root, COUNT = count])
;
```

```
; INPUTS:
;    fileFilter: Optional STRING denoting the file filter used in the search.
;                Any valid system command interpreter wildcards can be used.
;                If not supplied, one of the following is used:
;                   UNIX: '*'
;                  MACOS: '*'
;                    VMS: '*.*'
;                WINDOWS: '*.*'
;
; KEYWORD PARAMETERS:
;
;    RECURSE : Set this keyword to search recursively for matching files.
;      ROOT  : Set this keyword to a STRING denoting the directory from which
;              to start the search.  If not supplied, the current directory
;              is used.
;     COUNT  : A named variable into which the number of files found is placed.
;              If no files are found, a value of 0 is returned.
;
; OUTPUTS:
;    result: STRARR of matching files, or NULL string if no files are found.
;
; COMMON BLOCKS:
;    NONE
;
; SIDE EFFECTS:
;    None known
;
; RESTRICTIONS:
;    None known
;
; DEPENDENCIES:
;    NONE
;
; MODIFICATION HISTORY:
;    Written, 1998 May, Robert.Mallozzi@msfc.nasa.gov
;
;-
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

FUNCTION FINDFILES, fileSpec, RECURSE = recurse, ROOT = root, COUNT = count


   doRecurse = KEYWORD_SET (recurse)

   IF (N_ELEMENTS (root) NE 0) THEN BEGIN
     searchDir = root
   ENDIF ELSE BEGIN
     CD, CURRENT = searchDir
```

```
ENDELSE


CASE (STRUPCASE (!VERSION.OS_FAMILY)) OF

   'UNIX': BEGIN

      IF (N_ELEMENTS (fileSpec) EQ 0) THEN $
        fileSpec = '*'

      IF (doRecurse) THEN BEGIN

        command = 'find ' + searchDir + $
            ' -name "' + fileSpec + ""

      ENDIF ELSE BEGIN

        command = 'find ' + searchDir + $
            ' -maxdepth 1 -name "' + fileSpec + ""

      ENDELSE

      SPAWN, /SH, command, result
      END

   'VMS': BEGIN

      IF (N_ELEMENTS (fileSpec) EQ 0) THEN $
        fileSpec = '*.*'

      IF (doRecurse) THEN BEGIN

        command = STRMID (searchDir, 0, STRLEN (searchDir) - 1) + $
          '...]' + fileSpec

      ENDIF ELSE BEGIN

        command = fileSpec

      ENDELSE

      result = FINDFILE (command)
      END

   'MACOS': BEGIN

      IF (N_ELEMENTS (fileSpec) EQ 0) THEN $
        fileSpec = '*'
```

```
        result = FINDFILE (fileSpec)
        END

      'WINDOWS': BEGIN

        IF (N_ELEMENTS (fileSpec) EQ 0) THEN $
          fileSpec = '*.*'

        result = FINDFILE (fileSpec)
        END

      ELSE: MESSAGE, 'Unsupported operating system.'

  ENDCASE

  IF (result[0] EQ '') THEN BEGIN
    count = 0L
  ENDIF ELSE BEGIN
    count = N_ELEMENTS (result)
  ENDELSE


  RETURN, result

END

; end FINDFILES.PRO
```

--
Robert S. Mallozzi
http://cspar.uah.edu/~mallozzir/