Subject: Re: file date
Posted by Martin Schultz on Wed, 19 Nov 1997 08:00:00 GMT
View Forum Message <> Reply to Message

This is a multi-part message in MIME format.

----------167E2781446B
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit

Martin Schultz wrote:
>

thanks for your suggestions. Meanwhile I figured out a third way
(also Unix based) using the "find" command. Since what I am actually
interested in, is whether a given file is newer than my most recent
library compilation,
    find {filename} -newer {libname.tar.Z}
gives me exactly what I want.

Please find attached the routine that evolved out of this:
    update_library
It will automatically tell you which files have been added or which ones
are updated (useful if one tries to keep track of that e.g .on a web
page), and it will automatically copy all new/updated files into a
targetdirectory, tar and compress the stuff. Also useful in conjunction
with my distribute routine if you want to bundle a program package.

Regards,
Martin.

 -- -------
Dr. Martin Schultz
Department for Earth&Planetary Sciences, Harvard University
186 Pierce Hall, 29 Oxford St., Cambridge, MA-02138, USA

phone: (617)-496-8318
fax : (617)-495-4551

e-mail: mgs@io.harvard.edu
IDL-homepage: http://www-as.harvard.edu/people/staff/mgs/idl/
 -- -------

----------167E2781446B
Content-Type: text/plain; charset=us-ascii; name="update_library.pro"
Content-Transfer-Encoding: 7bit

Content-Disposition: inline; filename="update_library.pro"

```
 ;------------------------------------------------------------ --
;+
; NAME:
;       UPDATE_LIBRARY
;
; PURPOSE:
;       automatic update of IDL WWW library
;
; CATEGORY:
;       library handling routines
;
; CALLING SEQUENCE:
;       UPDATE_LIBRARY,TARGETDIR,TARNAME [,keywords]
;
; INPUTS:
;       TARGETDIR --> directory where WWW library resides
;         Usually a local directory, the contents of which is then
;         ftp'd to the www site (default: '~/IDL/lib-tools/').
;
;       TARNAME --> after copying all the required files, the library
;         is tar'd and compressed. TARNAME is the name of the uncompressed
;         file (default 'idl_lib.tar').
;
; KEYWORD PARAMETERS:
;       DIRS --> directories to search for files with SEARCHPATTERN.
;         This will only come to effect if no LISTFILE is given
;         (default ['~/IDL/tools/','~/IDL/tools3d']).
;
;       SEARCHPATTERN --> string array of filemasks which may contain
;         wildcards and determine the files to be included in the
;         library (default ['*.pro', '*.doc']). This
;         parameter is also used to obtain a list of files that are
;         already in the library and to determine the files to delete
;         if the PACKAGE option is set.
;
;       LISTFILE --> Name of a file that contains the names of all the
;         files to be included in the library (E.g. output from the
;         DISTRIBUTE routine, although this cannot be used directly).
;         In this case, no search for matching files is performed.
;
;       PACKAGE --> Normally, the individual files of the library will
;         remain in TARGETDIR together with the compressed tar file.
;         If the PACKAGE option is set, all individual files will be
;         deleted after creation of a new tar file. In order to retain
;         information about files once included in the library, the
;         file 'dirlist.old' is created before deleting the files.
```

```
;         Normally this file is created at the beginning of program
;         execution.
;
;       NO_ACTION --> DO not copy any files, create a tar file or compress
;         this. This will only produce a list of all the files that
;         would be included in the library together with their status
;         label ('NEW' or 'UPDATE', 'OLD' is represented byu blanks)
;
; OUTPUTS:
;       Unless the NO_ACTION keyword is set, the TARGETDIR will contain
;       a compressed tar file and unless the PACKAGE option is set, it
;       will also contain a copy of all individual files that make up the
;       library.
;
; SUBROUTINES:
;
; REQUIREMENTS:
;       Requires function STRRIGHT.
;
; NOTES:
;       This routine uses SPAWN extensively. The spawned commands are
;       UNIX specific.
;
;       The file status 'UPDATED' is obtained by comparing the filedate
;       of all files that are not 'NEW' with the date of the compressed
;       tar file using the Unix find command.
;
; EXAMPLE:
;           UPDATE_LIBRARY
;
;       will collect all '*.pro', and '*.doc' files in the directories
;       '~/IDL/tools', and '~/IDL/tools3d' , copy them into the
;       default target directory '~/IDL/lib-tools', tar them and compress
;       the tar file 'idl-lib.tar'.
;
;       In order to create a library with all '*.pro' and '*.sav' files
;       of all subdirectories of '~/IDL/' in the target directory 'lib-all'
;       type:
;
;           DIRS = STR_SEP(EXPAND_PATH('+~/IDL'),':')
;           SEARCH = ['*.pro','*.sav']
;           UPDATE_LIBRARY,'~/lib-all/','all_idl.tar',dirs=DIRS, $
;              searchpattern=SEARCH
;
;       In order to pack together all the files needed to run program
;       TEST.PRO, proceed as follows:
;
;         (if you are in IDL, exit and come back in)
```

```
;            JOURNAL,'test.files'
;            DISTRIBUTE,'test'
;            JOURNAL
;
;          (now edit your journal file 'test.files', keeping only the
;          names of the files that belong to the test.pro package, and
;          maybe add documentation and/or example files that you would
;          like to include in the library and create directory ~/lib-test)
;
;            UPDATE_LIBRARY,'~/lib-test','test.tar',listfile='test.files' , $
;              SEARCHPATTERN=['*.pro','*.doc','*.data'],/PACKAGE
;
;       Note, that the filemasks in SEARCHPATTERN will be used to
;       obtain the file status, and to find the files to be deleted after
;       updating the library. They are not necessary to include these
;       files in the library. This is completely controlled by the entries
;       of listfile.
;
;       If you want to collect all '*.data' files from a list of directories
;       in a library without keeping a copy of the individual files try:
;
;            dirlist=['~/data/','~/data/aircraft/','~/data/surface/']
;            UPDATE_LIBRARY,dirlist,'all_data.tar',/package, $
;              searchpattern='*.data'
;
;
; MODIFICATION HISTORY:
;       mgs, 19 Nov 1997: VERSION 1.00
;
;-
; Copyright (C) 1997, Martin Schultz, Harvard University
; This software is provided as is without any warranty
; whatsoever. It may be freely used, copied or distributed
; for non-commercial purposes. This copyright notice must be
; kept with any copy of this software. If this software shall
; be used commercially or sold as part of a larger package,
; please contact the author to arrange payment.
; Bugs and comments should be directed to mgs@io.harvard.edu
; with subject "IDL routine update_library"
 ;-------------------------------------------------------------- --


pro update_library,targetdir,tarname,  $
     dirs=dirs,searchpattern=searchpattern,listfile=listfile, $
     package=package,no_action=no_action


; set defaults for targetdir and tarname
```

```
if (n_elements(targetdir) le 0) then targetdir='~/IDL/lib-tools/'
if (strright(targetdir) ne '/') then targetdir = targetdir+'/'

if (n_elements(tarname) le 0) then tarname='idl_lib.tar'

print,'-----------------------------------'
print,'  update_library  : ',tarname,' in ',targetdir
print,'-----------------------------------'
print

; list of directories to be searched for new files
    if(n_elements(dirs) le 0) then $
       dirs = ['~/IDL/tools/','~/IDL/tools3d/']

    for i=0,n_elements(dirs)-1 do $
       if (strright(dirs(i)) ne '/') then dirs(i) = dirs(i)+'/'

; list of file masks to be searched for
    if(n_elements(searchpattern) le 0) then $
       searchpattern = [ '*.pro','*.doc' ]

; create blank delimited string with file masks
    searchstring = string(searchpattern,format="(255(' ',A,:))" )


; get current status of library
    cd,targetdir

    if(not keyword_set(package)) then begin
       cmd = 'ls'+searchstring+' > dirlist.old'
       print,'SPAWN: ',cmd
       spawn,cmd
    endif else $
       if(not file_exist('dirlist.old')) then begin
          print,'*** WARNING: dirlist.old does not exist ! ***'
          spawn,'touch dirlist.old'
       endif

    openr,ilun,'dirlist.old',/get_lun
    char=''
    old_files = ''
    while (not eof(ilun)) do begin
       readf,ilun,char
       old_files = [ old_files, char ]
    endwhile
    close,ilun
    ; NOTE that first element of old_files is dummy
```

```
; create filelist by searching all defaults or read in listfile
   filelist = ''
   if (n_elements(listfile) gt 0) then begin
      if(not file_exist(listfile,path=!path,full=full)) then begin
         print,'*** ERROR: Cannot find listfile '+listfile+' ! ***'
         return
      endif
      openr,ilun,full
      while (not eof(ilun)) do begin
         readf,ilun,char
         filelist = [ filelist, char ]
      endwhile
      close,ilun
   endif else begin    ; no listfile given
   ; loop through directories and find all files with all masks
      for i=0,n_elements(dirs)-1 do begin
         for j=0,n_elements(searchpattern)-1 do begin
            smask = dirs(i)+searchpattern(j)
            thisres = findfile(smask)
            filelist = [ filelist, thisres ]
         endfor
      endfor
   endelse

; remove all empty entries
   i = where(strlen(filelist) gt 1)
   if (i(0) ge 0) then filelist = filelist(i)  $
   else message,'No valid files !'

print,'filelist contains ',n_elements(filelist),' entries.'
; loop through all files of filelist, see if they exist and get full path
   for i=0,n_elements(filelist)-1 do  $
      if (file_exist(filelist(i),path=dirs,full=full)) then $
          filelist(i) = full $
      else $
          print,'*** WARNING: Cannot find file '+filelist(i)+' ! ***'

; sort filelist
   filelist = filelist(sort(filelist))



; check for new files and updates
; NEW FILES: Assume that contents of old_files represents status of old lib.
;   and check which files of filelist are not in there
print,'Checking for new files ...'

   filestatus = intarr(n_elements(filelist))
```

```
    for i=0,n_elements(filelist)-1 do begin
      test = extract_filename(filelist(i))
      ind = where(old_files eq test)
      if (ind(0) lt 0) then $
        filestatus(i) = 1    ; new file
    endfor

; UPDATED FILES: Use the unix find command for each entry of filelist
; which is not a new file and see if it is newer than last tarname.Z file
; Save the results in dirlist.new and read this file.
print,'Checking for updated files ...'

    spawn,'rm dirlist.new'
    spawn,'touch dirlist.new'
    if (file_exist(tarname+'.Z')) then begin
      for i=0,n_elements(filelist)-1 do begin
         cmd = 'find '+filelist(i)+' -newer '+tarname+'.Z >> dirlist.new'
         if (not filestatus(i)) then spawn,cmd
      endfor
    endif

    openr,ilun,'dirlist.new'
    updates = ''
    while (not eof(ilun)) do begin
      readf,ilun,char
      updates = [ updates, char ]
    endwhile
    close,ilun
    ; NOTE: first entry is dummy
    for i=0,n_elements(updates)-1 do $
      updates(i) = extract_filename(updates(i))

    for i=0,n_elements(filelist)-1 do begin
      test = extract_filename(filelist(i))
      ind = where(updates eq test)
      if (ind(0) ge 0) then $
        filestatus(i) = 2    ; updated file
    endfor

; print results of filestatus
    openw,ilun,'dirlist.new'
    for i=0,n_elements(filelist)-1 do begin
      if (filestatus(i) eq 0) then prefix = '        '
      if (filestatus(i) eq 1) then prefix = '    NEW '
      if (filestatus(i) eq 2) then prefix = ' UPDATE '
      print,prefix,extract_filename(filelist(i))
      printf,ilun,prefix,extract_filename(filelist(i))
```

```
    endfor
    close,ilun

    free_lun,ilun

; this is it if keyword NO_ACTION is set

    if (keyword_set(NO_ACTION)) then goto,the_end


; copy all files into target directory
print,'Copying ...'
for i=0,n_elements(filelist)-1 do begin
   if (filestatus(i) gt 0) then begin
      cmd = 'cp -p '+filelist(i)+' '+targetdir+extract_filename(filelist(i))
      spawn,cmd
   endif
endfor


; create "new" dirlist.old file if package option is set
    if(keyword_set(package)) then begin
       cmd = 'ls'+searchstring+' > dirlist.old'
       print,'SPAWN: ',cmd
       spawn,cmd
    endif



; tar and compress all files
   print,'Executing ...'

   cmd = 'tar -cf '+tarname+string(searchpattern,format="(256(' ',A,:))")
   print,cmd
   spawn,cmd

   cmd = 'compress -f '+tarname
   print,cmd
   spawn,cmd

; remove all single files if package option is set
   if(keyword_set(package)) then begin
       cmd = 'rm'+searchstring
       print,'SPAWN: ',cmd
       spawn,cmd
    endif

; ftp
```

; (maybe later) ...

the_end:
print,'Done.'
print

return
end

----------167E2781446B--

Subject: Re: file date
Posted by R. Bauer on Wed, 19 Nov 1997 08:00:00 GMT
View Forum Message <> Reply to Message

Martin Schultz wrote:

> Hi,
>
> is there any way to figure out the modification date of a given file in
> IDL ? I searched the manual but couldn't find anything. Unfortunately
> there is no such field in FSTAT. What I am trying to do is put together
> a small utitlity which would perform semi-automatic updates of my
> website library. Has anyone done such things before ?

Yes,

This is my true idl function. But it will only work I believe on unix
Systems.
I'll be lucky if someone could add the posibilities for PC's.

The trick is that's I use findfile whith parameters allowed by "ls"!

;

```
This
;       routine is provided as is without any express or implied
warranties
;       whatsoever.
;+
; NAME:
; get_date_of_file
;
; PURPOSE:
;   This function reads the modification time stamp from a given path and
file
;
; CATEGORY:
;   CASE_TOOLS
;
;
; CALLING SEQUENCE:
;   result=get_date_of_file(path,file_name)
;
;
; INPUTS:
;   path:       the path where the file is in
;   file_name: the name of the file
;
;
; OUTPUTS:
; This function returns the time stamp of the give file
;
;
; EXAMPLE:
; result=get_date_of_file('/','test')
;
;
; MODIFICATION HISTORY:
;  Written by: R.Bauer (ICG-1), Nov. 16 1997
;-


function get_date_of_file,dir, file_name

case strlowcase(!version.os) of
'aix' : slash_s = '/'
'linux' : slash_s ='/'
'windows' : slash_s ='\'
else:
endcase

openr,10,dir+slash_s+file_name,error=err
```

```
close,10
if err ne 0 then return,-1

info= str_sep(strcompress((findfile('-Fdl '+dir+slash_s+file_name))(0)),'
')

n_info=n_elements(info)-1

only_time=info(n_info-3:n_info-1)

a_year=(str_sep(systime(0),' '))(4)

if strpos(only_time(2),':') gt -1 then only_time(2)=a_year
only_time=arr2string(only_time,sep='-')
return,only_time

end
```

R.Bauer

--
R.Bauer

Institut fuer Stratosphaerische Chemie (ICG-1)
Forschungszentrum Juelich
email: R.Bauer@fz-juelich.de

Subject: Re: file date
Posted by Christian Soeller on Wed, 19 Nov 1997 08:00:00 GMT
View Forum Message <> Reply to Message

Martin Schultz <mgs@io.harvard.edu> writes:

> Hi,
> is there any way to figure out the modification date of a given file in
> IDL ? I searched the manual but couldn't find anything. Unfortunately
> there is no such field in FSTAT. What I am trying to do is put together
> a small utitlity which would perform semi-automatic updates of my
> website library. Has anyone done such things before ?

I am not aware of a function inside idl that would do this. However, if
you're on a unix system you should have the 'stat' command somewhere that,
on my system, prints the file modification time in seconds since 1970
if invoked as

stat -mq filename

So all you have to do inside idl is

```
function modtime, filename

exists = findfile(filename)
if (exists(0) eq '') then message, 'file '+filename+' not found'

spawn, 'stat -mq '+filename, mtime

return, long(mtime)
end
```

For other invocations of stat see 'man stat'. The only drawback doing it like this is the 'bizarre' signal handling of IDL that can cause 'spawn' to hang (at least that was true with IDL4X). If you have no fear of C coding you could code a stat system call in C and pass the field(s) your interested in with a home built routine using 'call_external'.

Hope this helps,

Christian

---------- --------
Christian Soeller mailto: csoelle@sghms.ac.uk
St. Georges Hospital Medical School Dept. of Pharmacology
Cranmer Terrace London SW17 0RE

Subject: Re: file date
Posted by Christian Soeller on Thu, 20 Nov 1997 08:00:00 GMT
View Forum Message <> Reply to Message

Martin Schultz <mgs@io.harvard.edu> writes:

> Please find attached the routine that evolved out of this:
> update_library
> It will automatically tell you which files have been added or which ones
> are updated (useful if one tries to keep track of that e.g .on a web
> page), and it will automatically copy all new/updated files into a
> targetdirectory, tar and compress the stuff. Also useful in conjunction
> with my distribute routine if you want to bundle a program package.
>

You might hate me for this (and will probably disagree), but:

The job you describe can be solved much more compact with a perl
script that generates a makefile which does the rest when called
with make -f 'makefile' (and will be a lot faster on a large number
of files). IDL is just not so great for tasks like
this (which are very far from numeric computations).

Christian
