Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'find.1p' command

$ man find.1p

FIND(1P) POSIX Programmer's Manual FIND(1P)

PROLOG

This manual page is part of the POSIX Programmer's Manual. The Linux implementation of this interface may differ (consult the corresponding Linux manual page for details of Linux behavior), or the interface may not be implemented on Linux.

NAME

find ? find files

SYNOPSIS

find [-H|-L] path... [operand_expression...]

DESCRIPTION

The find utility shall recursively descend the directory hierarchy from each file specified by path, evaluating a Boolean expression composed of the primaries described in the OPERANDS section for each file en? countered. Each path operand shall be evaluated unaltered as it was provided, including all trailing <slash> characters; all pathnames for other files encountered in the hierarchy shall consist of the concate? nation of the current path operand, a <slash> if the current path oper? and did not end in one, and the filename relative to the path operand. The relative portion shall contain no dot or dot-dot components, no trailing <slash> characters, and only single <slash> characters between pathname components.

The find utility shall be able to descend to arbitrary depths in a file

hierarchy and shall not fail due to path length limitations (unless a path operand specified by the application exceeds {PATH_MAX} require? ments).

The find utility shall detect infinite loops; that is, entering a pre? viously visited directory that is an ancestor of the last file encoun? tered. When it detects an infinite loop, find shall write a diagnostic message to standard error and shall either recover its position in the hierarchy or terminate.

If a file is removed from or added to the directory hierarchy being searched it is unspecified whether or not find includes that file in its search.

OPTIONS

The find utility shall conform to the Base Definitions volume of POSIX.1?2017, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported by the implementation:

-H Cause the file information and file type evaluated for each symbolic link encountered as a path operand on the command line to be those of the file referenced by the link, and not the link itself. If the referenced file does not exist, the file information and type shall be for the link itself. File information and type for symbolic links encountered during the traversal of a file hierarchy shall be that of the link itself.

-L Cause the file information and file type evaluated for each symbolic link encountered as a path operand on the command line or encountered during the traversal of a file hierarchy to be those of the file referenced by the link, and not the link itself. If the referenced file does not exist, the file information and type shall be for the link itself.

Specifying more than one of the mutually-exclusive options -H and -L shall not be considered an error. The last option specified shall de? termine the behavior of the utility. If neither the -H nor the -L op? tion is specified, then the file information and type for symbolic

links encountered as a path operand on the command line or encountered during the traversal of a file hierarchy shall be that of the link it? self.

OPERANDS

The following operands shall be supported:

The first operand and subsequent operands up to but not including the first operand that starts with a '-', or is a '!' or a '(', shall be interpreted as path operands. If the first operand starts with a '-', or is a '!' or a '(', the behavior is unspecified. Each path operand is a pathname of a starting point in the file hierarchy.

The first operand that starts with a '-', or is a '!' or a '(', and all subsequent arguments shall be interpreted as an expression made up of the following primaries and operators. In the descriptions, wherever n is used as a primary argument, it shall be interpreted as a decimal integer optionally preceded by a <plus-sign> ('+') or <hyphen-minus> ('-'), as follows:

+n More than n.

n Exactly n.

-n Less than n.

The following primaries shall be supported:

-name pattern

The primary shall evaluate as true if the basename of the current pathname matches pattern using the pattern matching notation described in Section 2.13, Pattern Matching Nota? tion. The additional rules in Section 2.13.3, Patterns Used for Filename Expansion do not apply as this is a matching op? eration, not an expansion.

-path pattern

The primary shall evaluate as true if the current pathname matches pattern using the pattern matching notation described in Section 2.13, Pattern Matching Notation. The additional rules in Section 2.13.3, Patterns Used for Filename Expansion do not apply as this is a matching operation, not an expan?

sion.

-nouser The primary shall evaluate as true if the file belongs to a user ID for which the getpwuid() function defined in the Sys? tem Interfaces volume of POSIX.1?2017 (or equivalent) returns NULL.

-nogroup The primary shall evaluate as true if the file belongs to a group ID for which the getgrgid() function defined in the System Interfaces volume of POSIX.1?2017 (or equivalent) re? turns NULL.

-xdev The primary shall always evaluate as true; it shall cause find not to continue descending past directories that have a different device ID (st_dev, see the stat() function defined in the System Interfaces volume of POSIX.1?2017). If any -xdev primary is specified, it shall apply to the entire ex? pression even if the -xdev primary would not normally be evaluated.

-prune The primary shall always evaluate as true; it shall cause find not to descend the current pathname if it is a direc? tory. If the -depth primary is specified, the -prune primary shall have no effect.

-perm [-]mode

The mode argument is used to represent file mode bits. It shall be identical in format to the symbolic_mode operand de? scribed in chmod, and shall be interpreted as follows. To start, a template shall be assumed with all file mode bits cleared. An op symbol of '+' shall set the appropriate mode bits in the template; '-' shall clear the appropriate bits; '=' shall set the appropriate mode bits, without regard to the contents of the file mode creation mask of the process. The op symbol of '-' cannot be the first character of mode; this avoids ambiguity with the optional leading <hyphen-mi? nus>. Since the initial mode is all bits off, there are not any symbolic modes that need to use '-' as the first charac?

ter.

If the <hyphen-minus> is omitted, the primary shall evaluate as true when the file permission bits exactly match the value of the resulting template.

Otherwise, if mode is prefixed by a <hyphen-minus>, the pri? mary shall evaluate as true if at least all the bits in the resulting template are set in the file permission bits.

-perm [-]onum

If the <hyphen-minus> is omitted, the primary shall evaluate as true when the file mode bits exactly match the value of the octal number onum (see the description of the octal mode in chmod). Otherwise, if onum is prefixed by a <hyphen-mi? nus>, the primary shall evaluate as true if at least all of the bits specified in onum are set. In both cases, the behav? ior is unspecified when onum exceeds 07777.

-type c The primary shall evaluate as true if the type of the file is c, where c is 'b', 'c', 'd', 'l', 'p', 'f', or 's' for block special file, character special file, directory, symbolic link, FIFO, regular file, or socket, respectively.

-links n The primary shall evaluate as true if the file has n links.

-user uname

The primary shall evaluate as true if the file belongs to the user uname. If uname is a decimal integer and the getpwnam() (or equivalent) function does not return a valid user name, uname shall be interpreted as a user ID.

-group gname

The primary shall evaluate as true if the file belongs to the group gname. If gname is a decimal integer and the getgr? nam() (or equivalent) function does not return a valid group name, gname shall be interpreted as a group ID.

-size n[c]

The primary shall evaluate as true if the file size in bytes, divided by 512 and rounded up to the next integer, is n. If

n is followed by the character 'c', the size shall be in bytes.

-atime n The primary shall evaluate as true if the file access time subtracted from the initialization time, divided by 86400 (with any remainder discarded), is n.

-ctime n The primary shall evaluate as true if the time of last change of file status information subtracted from the initialization time, divided by 86400 (with any remainder discarded), is n.

-mtime n The primary shall evaluate as true if the file modification time subtracted from the initialization time, divided by 86400 (with any remainder discarded), is n.

-exec utility_name [argument ...] ;

-exec utility_name [argument ...] {} +

The end of the primary expression shall be punctuated by a <semicolon> or by a <plus-sign>. Only a <plus-sign> that im? mediately follows an argument containing only the two charac? ters "{}" shall punctuate the end of the primary expression. Other uses of the <plus-sign> shall not be treated as spe? cial.

If the primary expression is punctuated by a <semicolon>, the utility utility_name shall be invoked once for each pathname and the primary shall evaluate as true if the utility returns a zero value as exit status. A utility_name or argument con? taining only the two characters "{}" shall be replaced by the current pathname. If a utility_name or argument string con? tains the two characters "{}", but not just the two charac? ters "{}", it is implementation-defined whether find replaces those two characters or uses the string without change.

If the primary expression is punctuated by a <plus-sign>, the primary shall always evaluate as true, and the pathnames for which the primary is evaluated shall be aggregated into sets. The utility utility_name shall be invoked once for each set of aggregated pathnames. Each invocation shall begin after

the last pathname in the set is aggregated, and shall be com? pleted before the find utility exits and before the first pathname in the next set (if any) is aggregated for this pri? mary, but it is otherwise unspecified whether the invocation occurs before, during, or after the evaluations of other pri? maries. If any invocation returns a non-zero value as exit status, the find utility shall return a non-zero exit status.

An argument containing only the two characters "{}" shall be replaced by the set of aggregated pathnames, with each path? name passed as a separate argument to the invoked utility in the same order that it was aggregated. The size of any set of two or more pathnames shall be limited such that execution of the utility does not cause the system's {ARG_MAX} limit to be exceeded. If more than one argument containing the two char? acters "{}" is present, the behavior is unspecified.

The current directory for the invocation of utility_name shall be the same as the current directory when the find utility was started. If the utility_name names any of the special built-in utilities (see Section 2.14, Special Built-In Utilities), the results are undefined.

-ok utility_name [argument ...] ;

The -ok primary shall be equivalent to -exec, except that the use of a <plus-sign> to punctuate the end of the primary ex? pression need not be supported, and find shall request affir? mation of the invocation of utility_name using the current file as an argument by writing to standard error as described in the STDERR section. If the response on standard input is affirmative, the utility shall be invoked. Otherwise, the command shall not be invoked and the value of the -ok operand shall be false.

-print The primary shall always evaluate as true; it shall cause the current pathname to be written to standard output.

-newer file

The primary shall evaluate as true if the modification time of the current file is more recent than the modification time of the file named by the pathname file.

-depth The primary shall always evaluate as true; it shall cause descent of the directory hierarchy to be done so that all entries in a directory are acted on before the directory itself. If a -depth primary is not specified, all entries in a directory shall be acted on after the directory itself. If any -depth primary is specified, it shall apply to the entire expression even if the -depth primary would not normally be evaluated.

The primaries can be combined using the following operators (in order of decreasing precedence):

(expression)

True if expression is true.

! expression

Negation of a primary; the unary NOT operator.

expression [-a] expression

Conjunction of primaries; the AND operator is implied by the juxtaposition of two primaries or made explicit by the operational -a operator. The second expression shall not be evaluated if the first expression is false.

expression -o expression

Alternation of primaries; the OR operator. The second expression shall not be evaluated if the first expression is true.

If no expression is present, -print shall be used as the expression. Otherwise, if the given expression does not contain any of the primaries -exec, -ok, or -print, the given expression shall be effectively replaced by:

(given_expression) -print

The -user, -group, and -newer primaries each shall evaluate their respective arguments only once.

When the file type evaluated for the current file is a symbolic link,

the results of evaluating the -perm primary are implementation-defined.

STDIN

If the -ok primary is used, the response shall be read from the stan? dard input. An entire line shall be read as the response. Otherwise, the standard input shall not be used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of find:

LANG Provide a default value for the internationalization vari? ables that are unset or null. (See the Base Definitions vol? ume of POSIX.1?2017, Section 8.2, Internationalization Vari? ables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL If set to a non-empty string value, override the values of all the other internationalization variables.

LC_COLLATE

Determine the locale for the behavior of ranges, equivalence classes, and multi-character collating elements used in the pattern matching notation for the -n option and in the ex? tended regular expression defined for the yesexpr locale key? word in the LC_MESSAGES category.

LC_CTYPE This variable determines the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in argu? ments), the behavior of character classes within the pattern matching notation used for the -n option, and the behavior of character classes within regular expressions used in the ex? tended regular expression defined for the yesexpr locale key? word in the LC_MESSAGES category.

LC_MESSAGES

Determine the locale used to process affirmative responses, and the locale used to affect the format and contents of di?

agnostic messages and prompts written to standard error.

NLSPATH Determine the location of message catalogs for the processing of LC_MESSAGES.

PATH Determine the location of the utility_name for the -exec and -ok primaries, as described in the Base Definitions volume of POSIX.1?2017, Chapter 8, Environment Variables.

ASYNCHRONOUS EVENTS

Default.

STDOUT

The -print primary shall cause the current pathnames to be written to standard output. The format shall be:

"%s\n", <path>

STDERR

The -ok primary shall write a prompt to standard error containing at least the utility_name to be invoked and the current pathname. In the POSIX locale, the last non-<blank> in the prompt shall be '?'. The ex? act format used is unspecified.

Otherwise, the standard error shall be used only for diagnostic mes? sages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

0 All path operands were traversed successfully.

>0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

When used in operands, pattern matching notation, <semicolon>, <left-parenthesis>, and <right-parenthesis> characters are special to the

shell and must be quoted (see Section 2.2, Quoting).

The bit that is traditionally used for sticky (historically 01000) is specified in the -perm primary using the octal number argument form. Since this bit is not defined by this volume of POSIX.1?2017, applica? tions must not assume that it actually refers to the traditional sticky bit.

EXAMPLES

1. The following commands are equivalent:

   find .

   find . -print

   They both write out the entire directory hierarchy from the current directory.

2. The following command:

   find / \(-name tmp -o -name '*.xx' \) -atime +7 -exec rm {} \;

   removes all files named tmp or ending in .xx that have not been ac? cessed for seven or more 24-hour periods.

3. The following command:

   find . -perm -o+w,+s

   prints (-print is assumed) the names of all files in or below the current directory, with all of the file permission bits S_ISUID, S_ISGID, and S_IWOTH set.

4. The following command:

   find . -name SCCS -prune -o -print

   recursively prints pathnames of all files in the current directory and below, but skips directories named SCCS and files in them.

5. The following command:

   find . -print -name SCCS -prune

   behaves as in the previous example, but prints the names of the SCCS directories.

6. The following command is roughly equivalent to the -nt extension to test:

   if [-n "$(find file1 -prune -newer file2)"]; then

   printf %s\\n "file1 is newer than file2"

```
fi
```

7. The descriptions of -atime, -ctime, and -mtime use the terminology n ``86400 second periods (days)''. For example, a file accessed at 23:59 is selected by:

```
find . -atime -1 -print
```

at 00:01 the next day (less than 24 hours later, not more than one day ago); the midnight boundary between days has no effect on the 24-hour calculation.

8. The following command:

```
find . ! -name . -prune -name '*.old' -exec \
    sh -c 'mv "$@" ../old/' sh {} +
```

performs the same task as:

```
mv ./*.old ./.old ./.*.old ../old/
```

while avoiding an ``Argument list too long'' error if there are a large number of files ending with .old and without running mv if there are no such files (and avoiding ``No such file or directory'' errors if ./.old does not exist or no files match ./*.old or ./.*.old).

The alternative:

```
find . ! -name . -prune -name '*.old' -exec mv {} ../old/ \;
```

is less efficient if there are many files to move because it exe? cutes one mv command per file.

9. On systems configured to mount removable media on directories under /media, the following command searches the file hierarchy for files larger than 100000 KB without searching any mounted removable me? dia:

```
find / -path /media -prune -o -size +200000 -print
```

10. Except for the root directory, and "//" on implementations where "//" does not refer to the root directory, no pattern given to -name will match a <slash>, because trailing <slash> characters are ignored when computing the basename of the file under evaluation. Given two empty directories named foo and bar, the following com? mand:

find foo/// bar/// -name foo -o -name 'bar?*'

prints only the line "foo///".

RATIONALE

The -a operator was retained as an optional operator for compatibility with historical shell scripts, even though it is redundant with expres? sion concatenation.

The descriptions of the '-' modifier on the mode and onum arguments to the -perm primary agree with historical practice on BSD and System V implementations. System V and BSD documentation both describe it in terms of checking additional bits; in fact, it uses the same bits, but checks for having at least all of the matching bits set instead of hav? ing exactly the matching bits set.

The exact format of the interactive prompts is unspecified. Only the general nature of the contents of prompts are specified because:

* Implementations may desire more descriptive prompts than those used on historical implementations.
* Since the historical prompt strings do not terminate with <newline> characters, there is no portable way for another program to inter? act with the prompts of this utility via pipes.

Therefore, an application using this prompting option relies on the system to provide the most suitable dialog directly with the user, based on the general guidelines specified.

The -name file operand was changed to use the shell pattern matching notation so that find is consistent with other utilities using pattern matching.

The -size operand refers to the size of a file, rather than the number of blocks it may occupy in the file system. The intent is that the st_size field defined in the System Interfaces volume of POSIX.1?2017 should be used, not the st_blocks found in historical implementations.

There are at least two reasons for this:

1. In both System V and BSD, find only uses st_size in size calcula? tions for the operands specified by this volume of POSIX.1?2017. (BSD uses st_blocks only when processing the -ls primary.)

2. Users usually think of file size in terms of bytes, which is also the unit used by the ls utility for the output from the -l option. (In both System V and BSD, ls uses st_size for the -l option size field and uses st_blocks for the ls -s calculations. This volume of POSIX.1?2017 does not specify ls -s.)

The descriptions of -atime, -ctime, and -mtime were changed from the SVID description of n ``days'' to n being the result of the integer di? vision of the time difference in seconds by 86400. The description is also different in terms of the exact timeframe for the n case (versus the +n or -n), but it matches all known historical implementations. It refers to one 86400 second period in the past, not any time from the beginning of that period to the current time. For example, -atime 2 is true if the file was accessed any time in the period from 72 hours to 48 hours ago.

Historical implementations do not modify "{}" when it appears as a sub? string of an -exec or -ok utility_name or argument string. There have been numerous user requests for this extension, so this volume of POSIX.1?2017 allows the desired behavior. At least one recent implemen? tation does support this feature, but encountered several problems in managing memory allocation and dealing with multiple occurrences of "{}" in a string while it was being developed, so it is not yet re? quired behavior.

Assuming the presence of -print was added to correct a historical pit? fall that plagues novice users, it is entirely upwards-compatible from the historical System V find utility. In its simplest form (find direc? tory), it could be confused with the historical BSD fast find. The BSD developers agreed that adding -print as a default expression was the correct decision and have added the fast find functionality within a new utility called locate.

Historically, the -L option was implemented using the primary -follow. The -H and -L options were added for two reasons. First, they offer a finer granularity of control and consistency with other programs that walk file hierarchies. Second, the -follow primary always evaluated to

true. As they were historically really global variables that took ef?
fect before the traversal began, some valid expressions had unexpected
results. An example is the expression -print -o -follow. Because
-print always evaluates to true, the standard order of evaluation im?
plies that -follow would never be evaluated. This was never the case.
Historical practice for the -follow primary, however, is not consis?
tent. Some implementations always follow symbolic links on the command
line whether -follow is specified or not. Others follow symbolic links
on the command line only if -follow is specified. Both behaviors are
provided by the -H and -L options, but scripts using the current -fol?
low primary would be broken if the -follow option is specified to work
either way.
Since the -L option resolves all symbolic links and the -type l primary
is true for symbolic links that still exist after symbolic links have
been resolved, the command:

    find -L . -type l

prints a list of symbolic links reachable from the current directory
that do not resolve to accessible files.
A feature of SVR4's find utility was the -exec primary's + terminator.
This allowed filenames containing special characters (especially <new?
line> characters) to be grouped together without the problems that oc?
cur if such filenames are piped to xargs. Other implementations have
added other ways to get around this problem, notably a -print0 primary
that wrote filenames with a null byte terminator. This was considered
here, but not adopted. Using a null terminator meant that any utility
that was going to process find's -print0 output had to add a new option
to parse the null terminators it would now be reading.
The "-exec...{}+" syntax adopted was a result of IEEE PASC Interpreta?
tion 1003.2 #210. It should be noted that this is an incompatible
change to IEEE Std 1003.2?1992. For example, the following command
printed all files with a '-' after their name if they are regular
files, and a '+' otherwise:

    find / -type f -exec echo {} - ';' -o -exec echo {} + ';'

The change invalidates usage like this. Even though the previous stan?
dard stated that this usage would work, in practice many did not sup?
port it and the standard developers felt it better to now state that
this was not allowable.

FUTURE DIRECTIONS

None.

SEE ALSO

Section 2.2, Quoting, Section 2.13, Pattern Matching Notation, Section 2.14, Special Built-In Utilities, chmod, mv, pax, sh, test

The Base Definitions volume of POSIX.1?2017, Chapter 8, Environment Variables, Section 12.2, Utility Syntax Guidelines

The System Interfaces volume of POSIX.1?2017, fstatat(), getgrgid(), getpwuid()

COPYRIGHT