

find Quick Reference

File search by name, type, size, time, permissions, and actions

Basic Search

Running find

```
find . # list all files recursively
find /var/log # search from specific path
find . -name "*.txt" # find by name
find / -name "config" 2>/dev/null # suppress permission errors
find dir1 dir2 -name "*.py" # search multiple directories
```

Syntax Overview

find [path...] [tests] [actions]	General form — paths, then tests, then actions
path	Starting directory (default: current directory)
test	Condition to filter files (-name, -type, etc.)
action	What to do with matches (-print, -exec, etc.)
Default action	-print if no action is specified

Name Patterns

Name Matching

```
find . -name "*.log" # case-sensitive glob
find . -iname "readme*" # case-insensitive glob
find . -name "*.py" -o -name "*.js" # OR: Python or JS files
find . -path "*/src/*.ts" # match against full path
find . -regex '.*\.(py|js|)' # POSIX regex on full path
```

Name Tests

-name pattern	Basename matches shell glob (case-sensitive)
-iname pattern	Basename matches glob (case-insensitive)
-path pattern	Full path matches shell glob
-ipath pattern	Full path matches glob (case-insensitive)
-regex pattern	Full path matches regular expression
-iregex pattern	Full path matches regex (case-insensitive)

Type Filters

Filter by Type

```
find . -type f # regular files only
find . -type d # directories only
find . -type l # symbolic links
find . -type f -name "*.sh" # combine type + name
```

File Types

-type f	Regular file
-type d	Directory
-type l	Symbolic link
-type b	Block device
-type c	Character device
-type p	Named pipe (FIFO)
-type s	Socket
-empty	Empty file or directory

Size & Time

Size & Time Examples

```
find . -size +100M # larger than 100 MB
find . -size -1k # smaller than 1 KB
find . -mtime -7 # modified in last 7 days
find . -mmin -30 # modified in last 30 minutes
find . -newer reference.txt # newer than reference file
```

Size & Time Tests

-size +/-Nc	Size in bytes (c), kilobytes (k), megabytes (M), gigabytes (G)
-mtime +/-N	Modified N*24 hours ago (+older, -newer)
-atime +/-N	Accessed N*24 hours ago
-ctime +/-N	Status changed N*24 hours ago
-mmin +/-N	Modified N minutes ago
-newer file	Modified more recently than file
-newermt date	Modified after date string (GNU)

Permissions

Permission Examples

```
find . -perm 644 # exact permissions: rw-r--r--
find . -perm -u+x # user has execute bit set
find . -perm /o+w # others have write (any match)
find . -user root # owned by root
find . -group www-data -type f # owned by group
```

Permission Tests

-perm mode	Exact permission match
-perm -mode	All specified bits are set
-perm /mode	Any specified bit is set
-user name	Owned by user (name or UID)
-group name	Owned by group (name or GID)
-nouser	No matching user in /etc/passwd
-nogroup	No matching group in /etc/group

Actions

Action Examples

```
find . -name "*.log" -print # print paths (default)
find . -name "*.tmp" -delete # delete matching files
find . -type f -ls # detailed listing
find . -name "*.txt" -print0 # null-delimited output
find . -type f -printf "%p %s\n" # custom format (GNU)
```

Action Reference

-print	Print path (newline-delimited)
-print0	Print path (null-delimited, safe for xargs)
-ls	Print file details (like ls -dils)
-delete	Delete matched files (implies -depth)
-printf format	Custom output format (GNU): %p path, %s size, %t time
-fprint file	Write paths to file
-quit	Exit after first match

Combining Tests

Logical Operators

```
find . -name "*.py" -type f # implicit AND
find . -name "*.py" -a -size +10k # explicit AND
find . -name "*.py" -o -name "*.js" # OR
find . ! -name "*.log" # NOT
find . \( -name "*.py" -o -name "*.js" \) -type f
```

Operator Reference

expr1 expr2 / expr1 -a expr2	AND — both must be true (default)
expr1 -o expr2	OR — either must be true
! expr / -not expr	NOT — negate the expression
\(expr \)	Group expressions (escape parens in shell)

Evaluation order Left to right; **-a** binds tighter than **-o**

Exec & Delete

Exec Examples

```
find . -name "*.sh" -exec chmod +x {} \;
find . -name "*.log" -exec rm {} +
find . -type f -exec grep -l "TODO" {} +
find . -name "*.bak" -ok rm {} \; # prompt before each
find . -name "*.tmp" -print0 | xargs -0 rm
```

Exec Reference

-exec cmd {} \;	Run cmd once per file ({} = file path)
-exec cmd {} +	Run cmd with multiple files at once (faster)
-ok cmd {} \;	Like -exec but prompts for confirmation
-execdir cmd {} \;	Run cmd from file's directory
xargs -0	Pair with -print0 for safe batch processing
-delete	Delete files; process deepest first

Depth & Pruning

Depth & Prune Examples

```
find . -maxdepth 1 -type f # current dir only
find . -mindepth 2 -name "*.py" # skip top-level
find . -name ".git" -prune -o -print # skip .git dirs
find . -depth -name "*.tmp" -delete # process children first
```

Depth Options

-maxdepth N	Descend at most N levels (0 = starting path only)
-mindepth N	Do not apply tests at levels less than N
-depth	Process directory contents before the directory itself
-prune	Do not descend into matched directory
-mount / -xdev	Do not cross filesystem boundaries
-follow / -L	Follow symbolic links

Common Patterns

One-Liners

```
find . -name "*.pyc" -delete # clean Python bytecode
find . -type f -size 0 -delete # remove empty files
find . -mtime +30 -name "*.log" -delete # purge old logs
find . -type f -name "*.md" | wc -l # count Markdown files
find . -type d -empty -delete # remove empty dirs
```

Recipes

Find largest files	find . -type f -printf '%s %p\n' sort -rn head
Find duplicates by name	find . -type f awk -F/'{print \$NF}' sort uniq -d
Rename extension	find . -name '*.txt' -exec rename 's/.txt/.md/' {} +
Find broken symlinks	find . -xtype l
Archive recent files	find . -mtime -7 -print0 tar czf recent.tar.gz --null -T -
Search code files	find . -name '*.py' -exec grep -l 'pattern' {} +