# fd Fast file finder

## CONCEPTS

fd searches names, not contents, and skips hidden files and anything in .gitignore by default. .gitignore is only honored inside a git repo. If a file you know exists is missing, try fd -u. Debian/Ubuntu install the binary as fdfind.

**Walk tree** current dir
↓ skip hidden + ignored
**Match name** regex, smart case
↓ -e -t -S filters
**Filter** type, size, age
↓ -x / -X
**Print or run** parallel

Smart case: an all-lowercase pattern ignores case, any capital makes it case-sensitive. Force it with -s or -i.

## SEARCHING

**fd**
List everything below the current directory

**fd** *<pattern>*
Names matching a regex

**fd** *<pattern>* *<dir>* [*<dir>*]
Search in given directories

**fd -g** '*<glob>*'
Match with a glob pattern

**fd -p** '*<regex>*'
Match against the full path

**fd -s** *<pattern>*
Force case-sensitive

**fd -i** *<pattern>*
Force case-insensitive

**fd -F** *<string>*
Treat pattern as a literal string

**fd --and** *<pattern>* *<pattern>*
Name must match both patterns

## FILTERS

**fd -e** *<ext>*
Only this extension

**fd -e** *<ext>* *<pattern>*
Extension plus name pattern

## FILTERS (cont.)

**fd -t f**
Regular files only

**fd -t d**
Directories only

**fd -t l**
Symlinks only

**fd -t x**
Executable files only

**fd -t e**
Empty files or directories

**fd -d** *<n>* *<pattern>*
Limit search depth

**fd -S +***<size>*
Files larger than size, e.g. +10M

**fd --changed-within** *<dur>*
Modified recently, e.g. 2d or 1h

**fd --changed-before** *<dur>*
Modified before a date or longer ago than a duration

**fd -E** '*<glob>*'
Exclude matching paths

**fd -L**
Follow symlinks

## HIDDEN AND IGNORED

**fd -H** *<pattern>*
Include hidden files

**fd -I** *<pattern>*
Do not respect ignore files

**fd -u** *<pattern>*
Unrestricted: same as -HI

**~/.config/fd/ignore**
Global ignore file (gitignore syntax)

**.fdignore**
Per-project ignore file, fd only

## OUTPUT

**fd -a** *<pattern>*
Print absolute paths

**fd -l** *<pattern>*
Detailed listing, like ls -l

**fd -0** *<pattern>*
NUL-separated output

**fd -c never** *<pattern>*
Turn colors off

## RUNNING COMMANDS

**fd -e** *<ext>* **-x** *<cmd>*
Run cmd once per result, in parallel

**fd -e** *<ext>* **-X** *<cmd>*
Run cmd once with all results

**fd -x** *<cmd>* **{}**
Placeholder: full path

**fd -x** *<cmd>* **{/}**
Placeholder: file name

**fd -x** *<cmd>* **{.}**
Placeholder: path without extension

**fd -x** *<cmd>* **{//}**
Placeholder: parent directory

**fd -x** *<cmd>* **{/.}**
Placeholder: name without extension

**fd -j** *<n>* **-x** *<cmd>*
Set number of parallel jobs

Without a placeholder, {} is appended automatically. Quote the whole command if it holds pipes or redirects, e.g. -x sh -c '...'.

## FD VS FIND

### Equivalents

| find ... | fd ... |
|---|---|
| -iname '*foo*' | fd foo |
| -name '*.md' | fd -e md |
| -type f | fd -t f |
| -mtime -2 | --changed-within 2d |
| -exec c {} \; | fd -x c |
| -exec c {} + | fd -X c |

fd is faster and has saner defaults, but find still wins for complex boolean tests like -perm, -newer and -prune. fd -u is the fair comparison to find.

## RECIPES

### Convert all PNGs to WebP

```
fd -e png -x magick {} {.}.webp
```

### Grep only some file types

```
fd -e cpp -e h -X rg 'std::cout'
```

### Delete stray files, with a prompt

```
fd -H '^\.DS_Store$' -t f -X rm -i
```