Rocky Enterprise Linux 9.2 Manual Pages on command 'git-grep.1'

C:\>man git-grep.1

GIT-GREP(1) Git Manual GIT-GREP(1)

NAME

git-grep - Print lines matching a pattern

SYNOPSIS

git grep [-a | --text] [-I] [--textconv] [-i | --ignore-case] [-w | --word-regexp]

[-v | --invert-match] [-h|-H] [--full-name]

[-E | --extended-regexp] [-G | --basic-regexp]

[-P | --perl-regexp]

[-F | --fixed-strings] [-n | --line-number] [--column]

[-l | --files-with-matches] [-L | --files-without-match]

[(-O | --open-files-in-pager) [<pager>]]

[-z | --null]

[-o | --only-matching] [-c | --count] [--all-match] [-q | --quiet]

[--max-depth <depth>] [--[no-]recursive]

[--color[=<when>] | --no-color]

[--break] [--heading] [-p | --show-function]

[-A <post-context>] [-B <pre-context>] [-C <context>]

[-W | --function-context]

[--threads <num>]

[-f <file>] [-e] <pattern>

[--and|--or|--not|(|)|-e <pattern>...]

[--recurse-submodules] [--parent-basename <basename>]

[[--[no-]exclude-standard] [--cached | --no-index | --untracked] | <tree>...]

[--] [<pathspec>...]

DESCRIPTION

Look for specified patterns in the tracked files in the work tree, blobs registered in the index file, or blobs in given tree objects. Patterns are lists of one or more search expressions separated by newline characters. An empty string as search expression matches all lines.

CONFIGURATION

grep.lineNumber

If set to true, enable -n option by default.

grep.column

If set to true, enable the --column option by default.

grep.patternType

Set the default matching behavior. Using a value of basic, extended, fixed, or perl will enable the --basic-regexp, --extended-regexp, --fixed-strings, or --perl-regexp option accordingly, while the value default will return to the default matching behavior.

grep.extendedRegexp

If set to true, enable --extended-regexp option by default. This option is ignored when the grep.patternType option is set to a value other than default.

grep.threads

Number of grep worker threads to use. If unset (or set to 0), 8 threads are used by default (for now).

grep.fullName

If set to true, enable --full-name option by default.

grep.fallbackToNoIndex

If set to true, fall back to git grep --no-index if git grep is executed outside of a git repository. Defaults to false.

OPTIONS

--cached

Instead of searching tracked files in the working tree, search blobs registered in the index file.

--no-index

Search files in the current directory that is not managed by Git.

--untracked

In addition to searching in the tracked files in the working tree, search also in untracked files.

--no-exclude-standard

Also search in ignored files by not honoring the .gitignore mechanism. Only useful with --untracked.

--exclude-standard

Do not pay attention to ignored files specified via the .gitignore mechanism. Only useful when searching files in the current directory with --no-index.

--recurse-submodules

Recursively search in each submodule that has been initialized and checked out in the repository. When used in combination with the <tree> option the prefix of all submodule output will be the name of the parent project?s <tree> object. This option has no effect if --no-index is given.

-a, --text

Process binary files as if they were text.

--textconv

Honor textconv filter settings.

--no-textconv

Do not honor textconv filter settings. This is the default.

-i, --ignore-case

Ignore case differences between the patterns and the files.

-I

Don?t match the pattern in binary files.

--max-depth <depth>

For each <pathspec> given on command line, descend at most <depth> levels of directories. A value of -1 means no limit. This option is ignored if <pathspec> contains active wildcards. In other words if "a*" matches a directory named "a*", "*" is matched literally so --max-depth is still effective.

-r, --recursive

Same as --max-depth=-1; this is the default.

--no-recursive

Same as --max-depth=0.

-w, --word-regexp

Match the pattern only at word boundary (either begin at the beginning of a line, or preceded by a non-word character; end at the end of a line or followed by a non-word character).

-v, --invert-match

Select non-matching lines.

-h, -H

By default, the command shows the filename for each match. -h option is used to suppress this output. -H is there for completeness and does not do anything except it overrides -h given earlier on the command line.

--full-name

When run from a subdirectory, the command usually outputs paths relative to the current directory. This option forces paths to be output relative to the project top directory.

-E, --extended-regexp, -G, --basic-regexp

Use POSIX extended/basic regexp for patterns. Default is to use basic regexp.

-P, --perl-regexp

Use Perl-compatible regular expressions for patterns.

Support for these types of regular expressions is an optional compile-time dependency. If Git wasn?t compiled with support for them providing this option will cause it to die.

-F, --fixed-strings

Use fixed strings for patterns (don?t interpret pattern as a regex).

-n, --line-number

Prefix the line number to matching lines.

--column

Prefix the 1-indexed byte-offset of the first match from the start of the matching line.

-l, --files-with-matches, --name-only, -L, --files-without-match

Instead of showing every matched line, show only the names of files that contain (or do not contain) matches. For better compatibility with git diff, --name-only is a synonym for --files-with-matches.

-O[<pager>], --open-files-in-pager[=<pager>]

Open the matching files in the pager (not the output of grep). If the pager happens to be "less" or "vi", and the user specified only one pattern, the first file is positioned at the first match automatically. The pager argument is optional; if specified, it must be stuck to the option without a space. If pager is unspecified, the default pager will be used (see core.pager in git-config(1)).

-z, --null

Output \0 instead of the character that normally follows a file name.

-o, --only-matching

Print only the matched (non-empty) parts of a matching line, with each such part on a separate output line.

-c, --count

Instead of showing every matched line, show the number of lines that match.

--color[=<when>]

Show colored matches. The value must be always (the default), never, or auto.

--no-color

Turn off match highlighting, even when the configuration file gives the default to color output. Same as --color=never.

--break

Print an empty line between matches from different files.

--heading

Show the filename above the matches in that file instead of at the start of each shown line.

-p, --show-function

Show the preceding line that contains the function name of the match, unless the matching line is a function name itself. The name is determined in the same way as git diff works out patch hunk headers (see Defining a custom hunk-header in gitattributes(5)).

-<num>, -C <num>, --context <num>

Show <num> leading and trailing lines, and place a line containing -- between contiguous groups of matches.

-A <num>, --after-context <num>

Show <num> trailing lines, and place a line containing -- between contiguous groups of matches.

-B <num>, --before-context <num>

Show <num> leading lines, and place a line containing -- between contiguous groups of matches.

-W, --function-context

Show the surrounding text from the previous line containing a function name up to the one before the next function name, effectively showing the whole function in which the match was found.

--threads <num>

Number of grep worker threads to use. See grep.threads in CONFIGURATION for more information.

-f <file>

Read patterns from <file>, one per line.

Passing the pattern via <file> allows for providing a search pattern containing a \0.

Not all pattern types support patterns containing \0. Git will error out if a given pattern type can?t support such a pattern. The --perl-regexp pattern type when compiled against the PCRE v2 backend has the widest support for these types of patterns.

In versions of Git before 2.23.0 patterns containing \0 would be silently considered fixed. This was never documented, there were also odd and undocumented interactions between e.g. non-ASCII patterns containing \0 and --ignore-case.

In future versions we may learn to support patterns containing \0 for more search backends, until then we?ll die when the pattern type in question doesn?t support them.

-e

The next parameter is the pattern. This option has to be used for patterns starting with - and should be used in scripts passing user input to grep.

Multiple patterns are combined by or.

--and, --or, --not, (...)

Specify how multiple patterns are combined using Boolean expressions. --or is

the default operator. --and has higher precedence than --or. -e has to be used for all patterns.

--all-match

When giving multiple pattern expressions combined with --or, this flag is specified to limit the match to files that have lines to match all of them.

-q, --quiet

Do not output matched lines; instead, exit with status 0 when there is a match and with non-zero status when there isn?t.

<tree>...

Instead of searching tracked files in the working tree, search blobs in the given trees.

--

Signals the end of options; the rest of the parameters are <pathspec> limiters.

<pathspec>...

If given, limit the search to paths matching at least one pattern. Both leading paths match and glob(7) patterns are supported.

For more details about the <pathspec> syntax, see the pathspec entry in gitglossary(7).

EXAMPLES

git grep 'time_t' -- '*.[ch]'

Looks for time_t in all tracked .c and .h files in the working directory and its subdirectories.

git grep -e '#define' --and \(-e MAX_PATH -e PATH_MAX \)

Looks for a line that has #define and either MAX_PATH or PATH_MAX.

git grep --all-match -e NODE -e Unexpected

Looks for a line that has NODE or Unexpected in files that have lines that match both.

git grep solution -- :^Documentation

Looks for solution, excluding files in Documentation.

GIT

Part of the git(1) suite