

NAME

git-annotate - Annotate file lines with commit information

SYNOPSIS

git annotate [options] file [revision]

DESCRIPTION

Annotates each line in the given file with information from the commit which introduced the line. Optionally annotates from a given revision.

The only difference between this command and [git-blame\(1\)](#) is that they use slightly different output formats, and this command exists only for backward compatibility to support existing scripts, and provide a more familiar command name for people coming from other SCM systems.

OPTIONS

-b

Show blank SHA-1 for boundary commits. This can also be controlled via the `blame.blankboundary` config option.

--root

Do not treat root commits as boundaries. This can also be controlled via the `blame.showroot` config option.

--show-stats

Include additional statistics at the end of blame output.

-L <start>,<end>, -L :<regex>

Annotate only the given line range. May be specified multiple times. Overlapping ranges are allowed.

<start> and <end> are optional. “-L <start>” or “-L<start>,” spans from<start> to end of file. “-L ,<end>” spans from start of file to<end>.

<start> and <end> can take one of these forms:

- number

If <start> or <end> is a number, it specifies an absolute line number (lines count from 1).

- /regex/

This form will use the first line matching the given POSIX regex. If <start> is a regex, it will search from the end of the previous -L range, if any, otherwise from the start of file. If <start> is “^/regex/”, it will search from the start of file. If <end> is a regex, it will search starting at the line given by <start>.

- +offset or -offset

This is only valid for <end> and will specify a number of lines before or after the line given by <start>.

If “:<regex>” is given in place of <start> and <end>, it denotes the range from the first filename line that matches <regex>, up to the next filename line. “:<regex>” searches from the end of the previous -L range, if any, otherwise from the start of file. “^:<regex>” searches from the start of file.

-l

Show long rev (Default: off).

-t

Show raw timestamp (Default: off).

-S <revs-file>

Use revisions from revs-file instead of calling [git-rev-list\(1\)](#).

--reverse

Walk history forward instead of backward. Instead of showing the revision in which a line appeared, this shows the last revision in which a line has existed. This requires a range of revision like `START..END` where the path to blame exists in `START`.

`-p, --porcelain`

Show in a format designed for machine consumption.

`--line-porcelain`

Show the porcelain format, but output commit information for each line, not just the first time a commit is referenced. Implies `--porcelain`.

`--incremental`

Show the result incrementally in a format designed for machine consumption.

`--encoding=<encoding>`

Specifies the encoding used to output author names and commit summaries. Setting it to `none` makes blame output unconverted data. For more information see the discussion about encoding in the [git-log\(1\)](#) manual page.

`--contents <file>`

When `<rev>` is not specified, the command annotates the changes starting backwards from the working tree copy. This flag makes the command pretend as if the working tree copy has the contents of the named file (specify `-` to make the command read from the standard input).

`--date <format>`

The value is one of the following alternatives: `{relative,local,default,iso,rfc,short}`. If `--date` is not provided, the value of the `blame.date` config variable is used. If the `blame.date` config variable is also not set, the `iso` format is used. For more information, See the discussion of the `--date` option at [git-log\(1\)](#).

`-M|<num>|`

Detect moved or copied lines within a file. When a commit moves or copies a block of lines (e.g. the original file has A and then B, and the commit changes it to B and then A), the traditional *blame* algorithm notices only half of the movement and typically blames the lines that were moved up (i.e. B) to the parent and assigns blame to the lines that were moved down (i.e. A) to the child commit. With this option, both groups of lines are blamed on the parent by running extra passes of inspection.

`<num>` is optional but it is the lower bound on the number of alphanumeric characters that Git must detect as moving/copying within a file for it to associate those lines with the parent commit. The default value is 20.

`-C|<num>|`

In addition to `-M`, detect lines moved or copied from other files that were modified in the same commit. This is useful when you reorganize your program and move code around across files. When this option is given twice, the command additionally looks for copies from other files in the commit that creates the file. When this option is given three times, the command additionally looks for copies from other files in any commit.

`<num>` is optional but it is the lower bound on the number of alphanumeric characters that Git must detect as moving/copying between files for it to associate those lines with the parent commit. And the default value is 40. If there are more than one `-C` options given, the `<num>` argument of the last `-C` will take effect.

`-h`

Show help message.

SEE ALSO

[git-blame\(1\)](#)

GIT

Part of the [git\(1\)](#) suite