NAME

git-status - Show the working tree status

SYNOPSIS

git status [<options>...] [--] [<pathspec>...]

DESCRIPTION

Displays paths that have differences between the index file and the current HEAD commit, paths that have differences between the working tree and the index file, and paths in the working tree that are not tracked by Git (and are not ignored by **gitignore(5)**). The first are what you *would* commit by running git commit; the second and third are what you *could* commit by running *git add* before running git commit.

OPTIONS

-s, --short
: Give the output in the short-format.

-b, --branch
: Show the branch and tracking info even in short-format.

--porcelain
: Give the output in an easy-to-parse format for scripts. This is similar to the short output, but will remain stable across Git versions and regardless of user configuration. See below for details.

--long
: Give the output in the long-format. This is the default.

-u[<mode>], --untracked-files[=<mode>]
: Show untracked files.

  The mode parameter is optional (defaults to *all*), and is used to specify the handling of untracked files.

  The possible options are:
  - *no* - Show no untracked files.
  - *normal* - Shows untracked files and directories.
  - *all* - Also shows individual files in untracked directories.

    When -u option is not used, untracked files and directories are shown (i.e. the same as specifying normal), to help you avoid forgetting to add newly created files. Because it takes extra work to find untracked files in the filesystem, this mode may take some time in a large working tree. You can use no to have git status return more quickly without showing untracked files.

    The default can be changed using the status.showUntrackedFiles configuration variable documented in **git-config(1)**.

--ignore-submodules[=<when>]
: Ignore changes to submodules when looking for changes. <when> can be either none, untracked, dirty or all, which is the default. Using none will consider the submodule modified when it either contains untracked or modified files or its HEAD differs from the commit recorded in the superproject and can be used to override any settings of the *ignore* option in **git-config(1)** or **gitmodules(5)**. When untracked is used submodules are not considered dirty when they only contain untracked content (but they are still scanned for modified content). Using dirty ignores all changes to the work tree of submodules, only changes to the commits stored in the superproject are shown (this was the behavior before 1.7.0). Using all hides all changes to submodules (and suppresses the output of submodule summaries when the config option status.submodulesummary is set).

--ignored
: Show ignored files as well.

-z
: Terminate entries with NUL, instead of LF. This implies the --porcelain output format if no other format is given.

--column[=<options>], --no-column
: Display untracked files in columns. See configuration variable column.status for option syntax.--column and --no-column without options are equivalent to *always* and *never* respectively.

OUTPUT

The output from this command is designed to be used as a commit template comment. The default, long format, is designed to be human readable, verbose and descriptive. Its contents and format are subject to change at any time.

The paths mentioned in the output, unlike many other Git commands, are made relative to the current directory if you are working in a subdirectory (this is on purpose, to help cutting and pasting). See the status.relativePaths config option below.

Short Format

In the short-format, the status of each path is shown as

XY PATH1 -> PATH2

where PATH1 is the path in the HEAD, and the -> PATH2 part is shown only when PATH1 corresponds to a different path in the index/worktree (i.e. the file is renamed). The *XY* is a two-letter status code.

The fields (including the ->) are separated from each other by a single space. If a filename contains whitespace or other nonprintable characters, that field will be quoted in the manner of a C string literal: surrounded by ASCII double quote (34) characters, and with interior special characters backslash-escaped.

For paths with merge conflicts, X and *Y* show the modification states of each side of the merge. For paths that do not have merge conflicts, X shows the status of the index, and Y shows the status of the work tree. For untracked paths, XY are ??. Other status codes can be interpreted as follows:

- = unmodified
- *M* = modified
- *A* = added
- *D* = deleted
- *R* = renamed
- *C* = copied
- *U* = updated but unmerged

Ignored files are not listed, unless --ignored option is in effect, in which case XY are !!.

X Y Meaning

[MD] not updated
M [MD] updated in index
A [MD] added to index
D [M] deleted from index
R [MD] renamed in index
C [MD] copied in index
[MARC] index and work tree matches
[MARC] M work tree changed since index
[MARC] D deleted in work tree

D D unmerged, both deleted
A U unmerged, added by us

U D unmerged, deleted by them
U A unmerged, added by them
D U unmerged, deleted by us
A A unmerged, both added
U U unmerged, both modified

? ? untracked
! ! ignored

If -b is used the short-format status is preceded by a line

branchname tracking info

Porcelain Format

The porcelain format is similar to the short format, but is guaranteed not to change in a backwards-incompatible way between Git versions or based on user configuration. This makes it ideal for parsing by scripts. The description of the short format above also describes the porcelain format, with a few exceptions:

1. The user's color.status configuration is not respected; color will always be off.
2. The user's status.relativePaths configuration is not respected; paths shown will always be relative to the repository root.

There is also an alternate -z format recommended for machine parsing. In that format, the status field is the same, but some other things change. First, the -> is omitted from rename entries and the field order is reversed (e.g *from* -> *to* becomes *to from*). Second, a NUL (ASCII 0) follows each filename, replacing space as a field separator and the terminating newline (but a space still separates the status field from the first filename). Third, filenames containing special characters are not specially formatted; no quoting or backslash-escaping is performed.

CONFIGURATION

The command honors color.status (or status.color — they mean the same thing and the latter is kept for backward compatibility) and color.status.<slot> configuration variables to colorize its output.

If the config variable status.relativePaths is set to false, then all paths shown are relative to the repository root, not to the current directory.

If status.submodulesummary is set to a non zero number or true (identical to -1 or an unlimited number), the submodule summary will be enabled for the long format and a summary of commits for modified submodules will be shown (see --summary-limit option of **git-submodule(1)**). Please note that the summary output from the status command will be suppressed for all submodules when diff.ignoreSubmodules is set to *all* or only for those submodules where submodule.<name>.ignore=all. To also view the summary for ignored submodules you can either use the --ignore-submodules=dirty command line option or the *git submodule summary*command, which shows a similar output but does not honor these settings.

SEE ALSO

gitignore(5)

GIT

Part of the **git(1)** suite