



Linux Ubuntu 26.04 Manual Pages on command 'git-ls-tree.1'

\$ man git-ls-tree.1

GIT-LS-TREE(1)

Git Manual

GIT-LS-TREE(1)

NAME

git-ls-tree - List the contents of a tree object

SYNOPSIS

git ls-tree [-d] [-r] [-t] [-l] [-z]

[--name-only] [--name-status] [--object-only] [--full-name] [--full-tree] [--abbrev[=<n>]] [--format=<format>]

<tree-ish> [<path>...]

DESCRIPTION

Lists the contents of a given tree object, like what `/bin/ls -a` does in the current working directory. Note that:

- ? the behaviour is slightly different from that of `/bin/ls` in that the `<path>` denotes just a list of patterns to match, e.g. so specifying directory name (without `-r`) will behave differently, and order of the arguments does not matter.
- ? the behaviour is similar to that of `/bin/ls` in that the `<path>` is taken as relative to the current working directory. E.g. when you are in a directory `sub` that has a directory `dir`, you can run `git ls-tree -r HEAD dir` to list the contents of the tree (that is `sub/dir` in `HEAD`). You don't want to give a tree that is not at the root level (e.g. `git ls-tree -r HEAD:sub dir`) in this case, as that would result in asking for `sub/sub/dir` in the `HEAD` commit. However, the current working directory can be ignored by passing `--full-tree` option.

OPTIONS

<tree-ish>

Id of a tree-ish.

-d

Show only the named tree entry itself, not its children.

-r

Recurse into sub-trees.

-t

Show tree entries even when going to recurse them. Has no effect if -r was not passed.

-d implies -t.

-l, --long

Show object size of blob (file) entries.

-Z

\0 line termination on output and do not quote filenames. See OUTPUT FORMAT below for more information.

--name-only, --name-status

List only filenames (instead of the "long" output), one per line. Cannot be combined with --object-only.

--object-only

List only names of the objects, one per line. Cannot be combined with --name-only or --name-status. This is equivalent to specifying --format='%(objectname)', but for both this option and that exact format the command takes a hand-optimized codepath instead of going through the generic formatting mechanism.

--abbrev[=<n>]

Instead of showing the full 40-byte hexadecimal object lines, show the shortest prefix that is at least <n> hexdigits long that uniquely refers the object. Non default number of digits can be specified with --abbrev=<n>.

--full-name

Instead of showing the path names relative to the current working directory, show the full path names.

--full-tree

Do not limit the listing to the current working directory. Implies --full-name.

--format=<format>

A string that interpolates %(fieldname) from the result being shown. It also interpolates %% to %, and %xNN where NN are hex digits interpolates to character with hex code NN; for example %x00 interpolates to \0 (NUL), %x09 to \t (TAB) and %x0a to \n

(LF). When specified, `--format` cannot be combined with other format-altering options, including `--long`, `--name-only` and `--object-only`.

[<path>...]

When paths are given, show them (note that this isn't really raw pathnames, but rather a list of patterns to match). Otherwise implicitly uses the root level of the tree as the sole path argument.

OUTPUT FORMAT

The output format of `ls-tree` is determined by either the `--format` option, or other format-altering options such as `--name-only` etc. (see `--format` above).

The use of certain `--format` directives is equivalent to using those options, but invoking the full formatting machinery can be slower than using an appropriate formatting option.

In cases where the `--format` would exactly map to an existing option `ls-tree` will use the appropriate faster path. Thus the default format is equivalent to:

```
%(objectmode) %(objecttype) %(objectname)%x09%(path)
```

This output format is compatible with what `--index-info --stdin` of `git update-index` expects.

When the `-l` option is used, format changes to

```
%(objectmode) %(objecttype) %(objectname) %(objectsize:padded)%x09%(path)
```

Object size identified by <objectname> is given in bytes, and right-justified with minimum width of 7 characters. Object size is given only for blobs (file) entries; for other entries

- character is used in place of size.

Without the `-z` option, pathnames with "unusual" characters are quoted as explained for the configuration variable `core.quotePath` (see `git-config(1)`). Using `-z` the filename is output verbatim and the line is terminated by a NUL byte.

Customized format:

It is possible to print in a custom format by using the `--format` option, which is able to interpolate different fields using a `%(fieldname)` notation. For example, if you only care about the "objectname" and "path" fields, you can execute with a specific `--format` like

```
git ls-tree --format='%(objectname) %(path)' <tree-ish>
```

FIELD NAMES

Various values from structured fields can be used to interpolate into the resulting output.

For each outputting line, the following names can be used:

objectmode

The mode of the object.

objecttype

The type of the object (commit, blob or tree).

objectname

The name of the object.

objectsize[:padded]

The size of a blob object ("-" if it's a commit or tree). It also supports a padded format of size with "%(objectsize:padded)".

path

The pathname of the object.

GIT

Part of the git(1) suite

Git 2.53.0

03/02/2026

GIT-LS-TREE(1)