git-revert

git-revert(1) ==========

NAME ---- git-revert - Revert some existing commits

SYNOPSIS --------

> 'git revert' [--[no-]edit] [-n] [-m <parent-number>] [-s] [-S[<keyid>]]
> <commit>...
> 'git revert' (--continue | --skip | --abort | --quit)

DESCRIPTION -----------

Given one or more existing commits, revert the changes that the related patches introduce, and record some new commits that record them. This requires your working tree to be clean (no modifications from the HEAD commit).

Note: 'git revert' is used to record some new commits to reverse the effect of some earlier commits (often only a faulty one). If you want to throw away all uncommitted changes in your working directory, you should see linkgit:git-reset[1], particularly the `--hard` option. If you want to extract specific files as they were in another commit, you should see linkgit:git-restore[1], specifically the `--source` option. Take care with these alternatives as both will discard uncommitted changes in your working directory.

See "Reset, restore and revert" in linkgit:git[1] for the differences between the three commands.

OPTIONS ------- <commit>...:: Commits to revert. For a more complete list of ways to spell commit names, see linkgit:gitrevisions[7]. Sets of commits can also be given but no traversal is done by default, see linkgit:git-rev-list[1] and its `--no-walk` option.

-e

--edit
With this option, 'git revert' will let you edit the commit message prior to committing the revert. This is the default if you run the command from a terminal.

-m parent-number

--mainline parent-number
Usually you cannot revert a merge because you do not know which side of the merge should be

considered the mainline. This option specifies the parent number (starting from 1) of the mainline and allows revert to reverse the change relative to the specified parent.

Reverting a merge commit declares that you will never want the tree changes brought in by the merge. As a result, later merges will only bring in tree changes introduced by commits that are not ancestors of the previously reverted merge. This may or may not be what you want.

See the revert-a-faulty-merge How-To for more details.

--no-edit
With this option, 'git revert' will not start the commit message editor.

--cleanup=<mode>
This option determines how the commit message will be cleaned up before being passed on to the commit machinery. See linkgit:git-commit[1] for more details. In particular, if the '<mode>' is given a value of `scissors`, scissors will be appended to `MERGE_MSG` before being passed on in the case of a conflict.

-n

--no-commit
Usually the command automatically creates some commits with commit log messages stating which commits were reverted. This flag applies the changes necessary to revert the named commits to your working tree and the index, but does not make the commits. In addition, when this option is used, your index does not have to match the HEAD commit. The revert is done against the beginning state of your index.

This is useful when reverting more than one commits' effect to your index in a row.

-S[<keyid>]

--gpg-sign[=<keyid>]

--no-gpg-sign
GPG-sign commits. The `keyid` argument is optional and defaults to the committer identity; if specified, it must be stuck to the option without a space. `--no-gpg-sign` is useful to countermand both `commit.gpgSign` configuration variable, and earlier `--gpg-sign`.

-s

--signoff
Add a `Signed-off-by` trailer at the end of the commit message. See the signoff option in linkgit:git-commit[1] for more information.

--strategy=<strategy>
Use the given merge strategy. Should only be used once. See the MERGE STRATEGIES section in linkgit:git-merge[1] for details.

-X<option>

--strategy-option=<option>
Pass the merge strategy-specific option through to the merge strategy. See linkgit:git-merge[1] for details.

`--rerere-autoupdate`

`--no-rerere-autoupdate`
After the rerere mechanism reuses a recorded resolution on the current conflict to update the files in the working tree, allow it to also update the index with the result of resolution. `--no-rerere-autoupdate` is a good way to double-check what linkgit:git-rerere[1] did and catch potential mismerges, before committing the result to the index with a separate linkgit:git-add[1].

--reference
Instead of starting the body of the log message with "This reverts <full-object-name-of-the-commit-being-reverted>.", refer to the commit using "--pretty=reference" format (cf. linkgit:git-log[1]). The `revert.reference` configuration variable can be used to enable this option by default.

SEQUENCER SUBCOMMANDS ---------- include::sequencer.adoc[]

EXAMPLES -------- `git` `revert` `HEAD~3`::

```
Revert the changes specified by the fourth last commit in HEAD
and create a new commit with the reverted changes.
```

`git` `revert` `-n` `master` `5..master` `2`

```
Revert the changes done by commits from the fifth last commit
in master (included) to the third last commit in master
(included), but do not create any commit with the reverted
changes. The revert only modifies the working tree and the
index.
```

DISCUSSION ----------

While git creates a basic commit message automatically, it is *strongly* recommended to explain why the original commit is being reverted. In addition, repeatedly reverting reverts will result in increasingly unwieldy subject lines, for example ’Reapply "Reapply "<original-subject>""’. Please consider rewording these to be shorter and more unique.

CONFIGURATION ----------

Everything below this line in this section is selectively included from the linkgit:git-config[1] documentation. The content is the same as what’s found there:

revert.reference
Setting this variable to true makes `git` `revert` behave as if the `--reference` option is given.

SEE ALSO -------- linkgit:git-cherry-pick[1]

GIT --- Part of the linkgit:git[1] suite

□□□ #1
□ □□□ □□□ 2026-10-31 20:12:47 UTC
□ □□□ □□□ 2026-10-31 20:12:47 UTC