

**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 **git-reset**(1), particularly the **--hard** option. If you want to extract specific files as they were in another commit, you should see **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 **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 **gitrevisions**(7). Sets of commits can also be given but no traversal is done by default, see **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**<sup>[1]</sup> 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 **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>]  
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.
- s, --signoff  
Add Signed-off-by line at the end of the commit message. See the signoff option in **git-commit(1)** for more information.
- strategy=<strategy>  
Use the given merge strategy. Should only be used once. See the MERGE STRATEGIES section in **git-merge(1)** for details.
- X<option>, --strategy-option=<option>  
Pass the merge strategy-specific option through to the merge strategy. See **git-merge(1)** for details.
- rerere-autoupdate, --no-rerere-autoupdate  
Allow the rerere mechanism to update the index with the result of auto-conflict resolution if possible.

## SEQUENCER SUBCOMMANDS

- continue  
Continue the operation in progress using the information in **.git/sequencer**. Can be used to continue after resolving conflicts in a failed cherry-pick or revert.
- skip  
Skip the current commit and continue with the rest of the sequence.
- quit  
Forget about the current operation in progress. Can be used to clear the sequencer state after a failed cherry-pick or revert.
- abort  
Cancel the operation and return to the pre-sequence state.

## 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.

## SEE ALSO

**git-cherry-pick(1)**

## GIT

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

## NOTES

1. revert-a-faulty-merge How-To  
<file:///usr/share/doc/git/html/howto/revert-a-faulty-merge.html>