



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

\$ man git-mergetool.1

GIT-MERGETOOL(1)

Git Manual

GIT-MERGETOOL(1)

NAME

git-mergetool - Run merge conflict resolution tools to resolve merge conflicts

SYNOPSIS

git mergetool [--tool=<tool>] [-y | --[no-]prompt] [<file>...]

DESCRIPTION

Use git mergetool to run one of several merge utilities to resolve merge conflicts. It is typically run after git merge.

If one or more <file> parameters are given, the merge tool program will be run to resolve differences in each file (skipping those without conflicts). Specifying a directory will include all unresolved files in that path. If no <file> names are specified, git mergetool will run the merge tool program on every file with merge conflicts.

OPTIONS

-t <tool>, --tool=<tool>

Use the merge resolution program specified by <tool>. Valid values include emerge, gvimdiff, kdiff3, meld, vimdiff, and tortoisemerge. Run git mergetool --tool-help for the list of valid <tool> settings.

If a merge resolution program is not specified, git mergetool will use the configuration variable merge.tool. If the configuration variable merge.tool is not set, git mergetool will pick a suitable default.

You can explicitly provide a full path to the tool by setting the configuration variable mergetool.<tool>.path. For example, you can configure the absolute path to kdiff3 by setting mergetool.kdiff3.path. Otherwise, git mergetool assumes the tool is available in

\$PATH.

Instead of running one of the known merge tool programs, git mergetool can be customized to run an alternative program by specifying the command line to invoke in a configuration variable mergetool.<tool>.cmd.

When git mergetool is invoked with this tool (either through the -t or --tool option or the merge.tool configuration variable), the configured command line will be invoked with BASE set to the name of a temporary file containing the common base for the merge, if available; LOCAL set to the name of a temporary file containing the contents of the file on the current branch; REMOTE set to the name of a temporary file containing the contents of the file to be merged, and MERGED set to the name of the file to which the merge tool should write the result of the merge resolution.

If the custom merge tool correctly indicates the success of a merge resolution with its exit code, then the configuration variable mergetool.<tool>.trustExitCode can be set to true. Otherwise, git mergetool will prompt the user to indicate the success of the resolution after the custom tool has exited.

--tool-help

Print a list of merge tools that may be used with --tool.

-y, --no-prompt

Don't prompt before each invocation of the merge resolution program. This is the default if the merge resolution program is explicitly specified with the --tool option or with the merge.tool configuration variable.

--prompt

Prompt before each invocation of the merge resolution program to give the user a chance to skip the path.

-g, --gui

When git-mergetool is invoked with the -g or --gui option, the default merge tool will be read from the configured merge.guitool variable instead of merge.tool. If merge.guitool is not set, we will fallback to the tool configured under merge.tool. This may be autoselected using the configuration variable mergetool.guiDefault.

--no-gui

This overrides a previous -g or --gui setting or mergetool.guiDefault configuration and reads the default merge tool from the configured merge.tool variable.

-O<orderfile>

Process files in the order specified in the <orderfile>, which has one shell glob pattern per line. This overrides the diff.orderFile configuration variable (see git-config(1)). To cancel diff.orderFile, use -O/dev/null.

CONFIGURATION

Everything below this line in this section is selectively included from the git-config(1) documentation. The content is the same as what's found there:

mergetool.<tool>.path

Override the path for the given tool. This is useful in case your tool is not in the \$PATH.

mergetool.<tool>.cmd

Specify the command to invoke the specified merge tool. The specified command is evaluated in shell with the following variables available: BASE is the name of a temporary file containing the common base of the files to be merged, if available; LOCAL is the name of a temporary file containing the contents of the file on the current branch; REMOTE is the name of a temporary file containing the contents of the file from the branch being merged; MERGED contains the name of the file to which the merge tool should write the results of a successful merge.

mergetool.<tool>.hideResolved

Allows the user to override the global mergetool.hideResolved value for a specific tool.

See mergetool.hideResolved for the full description.

mergetool.<tool>.trustExitCode

For a custom merge command, specify whether the exit code of the merge command can be used to determine whether the merge was successful. If this is not set to true then the merge target file timestamp is checked, and the merge is assumed to have been successful if the file has been updated; otherwise, the user is prompted to indicate the success of the merge.

mergetool.meld.hasOutput

Older versions of meld do not support the --output option. Git will attempt to detect whether meld supports --output by inspecting the output of meld --help. Configuring mergetool.meld.hasOutput will make Git skip these checks and use the configured value instead. Setting mergetool.meld.hasOutput to true tells Git to unconditionally use the --output option, and false avoids using --output.

mergetool.meld.useAutoMerge

When the `--auto-merge` is given, meld will merge all non-conflicting parts automatically, highlight the conflicting parts, and wait for user decision. Setting `mergetool.meld.useAutoMerge` to true tells Git to unconditionally use the `--auto-merge` option with meld. Setting this value to auto makes git detect whether `--auto-merge` is supported and will only use `--auto-merge` when available. A value of false avoids using `--auto-merge` altogether, and is the default value.

`mergetool.<variant>.layout`

Configure the split window layout for vimdiff's `<variant>`, which is any of vimdiff, nvimdiff, gvimdiff. Upon launching git mergetool with `--tool=<variant>` (or without `--tool` if `merge.tool` is configured as `<variant>`), Git will consult `mergetool.<variant>.layout` to determine the tool's layout. If the variant-specific configuration is not available, vimdiff 's is used as fallback. If that too is not available, a default layout with 4 windows will be used. To configure the layout, see the BACKEND SPECIFIC HINTS section.

`mergetool.hideResolved`

During a merge, Git will automatically resolve as many conflicts as possible and write the \$MERGED file containing conflict markers around any conflicts that it cannot resolve; \$LOCAL and \$REMOTE normally are the versions of the file from before Git's conflict resolution. This flag causes \$LOCAL and \$REMOTE to be overwritten so that only the unresolved conflicts are presented to the merge tool. Can be configured per-tool via the `mergetool.<tool>.hideResolved` configuration variable. Defaults to false.

`mergetool.keepBackup`

After performing a merge, the original file with conflict markers can be saved as a file with a .orig extension. If this variable is set to false then this file is not preserved. Defaults to true (i.e. keep the backup files).

`mergetool.keepTemporaries`

When invoking a custom merge tool, Git uses a set of temporary files to pass to the tool. If the tool returns an error and this variable is set to true, then these temporary files will be preserved; otherwise, they will be removed after the tool has exited. Defaults to false.

`mergetool.writeToTemp`

Git writes temporary BASE, LOCAL, and REMOTE versions of conflicting files in the worktree by default. Git will attempt to use a temporary directory for these files when

set true. Defaults to false.

mergetool.prompt

Prompt before each invocation of the merge resolution program.

mergetool.guiDefault

Set true to use the merge.guitool by default (equivalent to specifying the --gui argument), or auto to select merge.guitool or merge.tool depending on the presence of a DISPLAY environment variable value. The default is false, where the --gui argument must be provided explicitly for the merge.guitool to be used.

TEMPORARY FILES

git mergetool creates *.orig backup files while resolving merges. These are safe to remove once a file has been merged and its git mergetool session has completed.

Setting the mergetool.keepBackup configuration variable to false causes git mergetool to automatically remove the backup files as files are successfully merged.

BACKEND SPECIFIC HINTS

vimdiff

Description

When specifying --tool=vimdiff in git mergetool Git will open Vim with a 4 windows layout distributed in the following way:

```
-----
|      |      |      |
| LOCAL | BASE | REMOTE |
|      |      |      |
-----
|                  |
|      MERGED      |
|                  |
-----
```

LOCAL, BASE and REMOTE are read-only buffers showing the contents of the conflicting file in specific commits ("commit you are merging into", "common ancestor commit" and "commit you are merging from" respectively)

MERGED is a writable buffer where you have to resolve the conflicts (using the other read-only buffers as a reference). Once you are done, save and exit Vim as usual (:wq) or, if you want to abort, exit using :cq.

Layout configuration

You can change the windows layout used by Vim by setting configuration variable `mergetool.vimdiff.layout` which accepts a string where the following separators have special meaning:

? `+` is used to "open a new tab"

? `,` is used to "open a new vertical split"

? `/` is used to "open a new horizontal split"

? `@` is used to indicate the file containing the final version after solving the conflicts. If not present, `MERGED` will be used by default.

The precedence of the operators is as follows (you can use parentheses to change it):

``@` > ``+` > `/` >`,``

Let's see some examples to understand how it works:

? `layout = "(LOCAL,BASE,REMOTE)/MERGED"`

This is exactly the same as the default layout we have already seen.

Note that `/` has precedence over `,` and thus the parenthesis are not needed in this case. The next layout definition is equivalent:

`layout = "LOCAL,BASE,REMOTE / MERGED"`

? `layout = "LOCAL,MERGED,REMOTE"`

If, for some reason, we are not interested in the BASE buffer.

```
-----  
|      |      |      |  
|      |      |      |  
| LOCAL | MERGED | REMOTE |  
|      |      |      |  
|      |      |      |  
-----
```

? `layout = "MERGED"`

Only the `MERGED` buffer will be shown. Note, however, that all the other ones are still loaded in vim, and you can access them with the "buffers" command.

```
-----  
|      |  
|      |  
|      MERGED      |  
|      |
```

```

|           |
|           |
-----

```

? layout = "@LOCAL,REMOTE"

When MERGED is not present in the layout, you must "mark" one of the buffers with an arobase (@). That will become the buffer you need to edit and save after resolving the conflicts.

```

-----
|           |           |
|           |           |
|           |           |
|  LOCAL    |  REMOTE    |
|           |           |
|           |           |
|           |           |
-----

```

? layout = "LOCAL,BASE,REMOTE / MERGED + BASE,LOCAL + BASE,REMOTE"

Three tabs will open: the first one is a copy of the default layout, while the other two only show the differences between (BASE and LOCAL) and (BASE and REMOTE) respectively.

```

-----
| <TAB #1> | TAB #2 | TAB #3 |   |
-----
|           |           |           |
|  LOCAL    |  BASE    |  REMOTE    |
|           |           |           |
-----
|           |           |
|           | MERGED   |
|           |           |
-----
| TAB #1 | <TAB #2> | TAB #3 |   |

```

	BASE	LOCAL

| TAB #1 | TAB #2 | <TAB #3> | |

	BASE	REMOTE

? layout = "LOCAL,BASE,REMOTE / MERGED + BASE,LOCAL + BASE,REMOTE + (LOCAL/BASE/REMOTE),MERGED"

Same as the previous example, but adds a fourth tab with the same information as the first tab, with a different layout.

| TAB #1 | TAB #2 | TAB #3 | <TAB #4> |

	LOCAL	

	BASE	MERGED

	REMOTE	

Note how in the third tab definition we need to use parentheses to make , have precedence over /.

Variants

Instead of `--tool=vimdiff`, you can also use one of these other variants:

? `--tool=gvimdiff`, to open gVim instead of Vim.

? `--tool=nvimdiff`, to open Neovim instead of Vim.

When using these variants, in order to specify a custom layout you will have to set configuration variables `mergetool.gvimdiff.layout` and `mergetool.nvimdiff.layout` instead of `mergetool.vimdiff.layout` (though the latter will be used as fallback if the variant-specific one is not set).

In addition, for backwards compatibility with previous Git versions, you can also append 1, 2 or 3 to either `vimdiff` or any of the variants (ex: `vimdiff3`, `nvimdiff1`, etc...) to use a predefined layout. In other words, using `--tool=[g|n]vimdiff<x>` is the same as using `--tool=[g|n]vimdiff` and setting configuration variable `mergetool.[g|n]vimdiff.layout` to...

? `<x>=1`: "@LOCAL, REMOTE"

? `<x>=2`: "LOCAL, MERGED, REMOTE"

? `<x>=3`: "MERGED"

Example: using `--tool=gvimdiff2` will open gvim with three columns (LOCAL, MERGED and REMOTE).

GIT

Part of the git(1) suite

Git 2.53.0

03/02/2026

GIT-MERGETOOL(1)