NAME

git-am - Apply a series of patches from a mailbox

SYNOPSIS

git am [--signoff] [--keep] [--[no-]keep-cr] [--[no-]utf8] [--no-verify]
[--[no-]3way] [--interactive] [--committer-date-is-author-date]
[--ignore-date] [--ignore-space-change | --ignore-whitespace]
[--whitespace=<option>] [-C<n>] [-p<n>] [--directory=<dir>]
[--exclude=<path>] [--include=<path>] [--reject] [-q | --quiet]
[--[no-]scissors] [-S[<keyid>]] [--patch-format=<format>]
[--quoted-cr=<action>]
[--empty=(stop|drop|keep)]
[(<mbox> | <Maildir>)...]
git am (--continue | --skip | --abort | --quit | --show-current-patch[=(diff|raw)] | --allow-empty)

DESCRIPTION

Splits mail messages in a mailbox into commit log message, authorship information and patches, and applies them to the current branch. You could think of it as a reverse operation of **git-format-patch**(1) run on a branch with a straight history without merges.

OPTIONS

(<mbox>|<Maildir>)...

The list of mailbox files to read patches from. If you do not supply this argument, the command reads from the standard input. If you supply directories, they will be treated as Maildirs.

-s, --signoff

Add a **Signed-off-by** trailer to the commit message, using the committer identity of yourself. See the signoff option in **git-commit**(1) for more information.

-k, --keep

Pass **-k** flag to *git mailinfo* (see **git-mailinfo**(1)).

--keep-non-patch

Pass **-b** flag to *git mailinfo* (see **git-mailinfo**(1)).

--[no-]keep-cr

With **--keep-cr**, call *git mailsplit* (see **git-mailsplit**(1)) with the same option, to prevent it from stripping CR at the end of lines. **am.keepcr** configuration variable can be used to specify the default behaviour. **--no-keep-cr** is useful to override **am.keepcr**.

-c, --scissors
: Remove everything in body before a scissors line (see **git-mailinfo**(1)). Can be activated by default using the **mailinfo.scissors** configuration variable.

--no-scissors
: Ignore scissors lines (see **git-mailinfo**(1)).

--quoted-cr=<action>
: This flag will be passed down to *git mailinfo* (see **git-mailinfo**(1)).

--empty=(stop|drop|keep)
: By default, or when the option is set to *stop*, the command errors out on an input e-mail message lacking a patch and stops into the middle of the current am session. When this option is set to *drop*, skip such an e-mail message instead. When this option is set to *keep*, create an empty commit, recording the contents of the e-mail message as its log.

-m, --message-id
: Pass the **-m** flag to *git mailinfo* (see **git-mailinfo**(1)), so that the Message-ID header is added to the commit message. The **am.messageid** configuration variable can be used to specify the default behaviour.

--no-message-id
: Do not add the Message-ID header to the commit message. **no-message-id** is useful to override **am.messageid**.

-q, --quiet
: Be quiet. Only print error messages.

-u, --utf8
: Pass **-u** flag to *git mailinfo* (see **git-mailinfo**(1)). The proposed commit log message taken from the e-mail is re-coded into UTF-8 encoding (configuration variable **i18n.commitEncoding** can be used to specify project's preferred encoding if it is not UTF-8).

  This was optional in prior versions of git, but now it is the default. You can use **--no-utf8** to override this.

--no-utf8
: Pass **-n** flag to *git mailinfo* (see **git-mailinfo**(1)).

-3, --3way, --no-3way

When the patch does not apply cleanly, fall back on 3-way merge if the patch records the identity of blobs it is supposed to apply to and we have those blobs available locally. **--no-3way** can be used to override am.threeWay configuration variable. For more information, see am.threeWay in **git-config**(1).

--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 **rerere** did and catch potential mismerges, before committing the result to the index with a separate **git add**.

--ignore-space-change, --ignore-whitespace, --whitespace=<option>, -C<n>, -p<n>, --directory=<dir>, --exclude=<path>, --include=<path>, --reject

These flags are passed to the *git apply* (see **git-apply**(1)) program that applies the patch.

--patch-format

By default the command will try to detect the patch format automatically. This option allows the user to bypass the automatic detection and specify the patch format that the patch(es) should be interpreted as. Valid formats are mbox, mboxrd, stgit, stgit-series and hg.

-i, --interactive

Run interactively.

-n, --no-verify

By default, the pre-applypatch and applypatch-msg hooks are run. When any of **--no-verify** or **-n** is given, these are bypassed. See also **githooks**(5).

--committer-date-is-author-date

By default the command records the date from the e-mail message as the commit author date, and uses the time of commit creation as the committer date. This allows the user to lie about the committer date by using the same value as the author date.

--ignore-date

By default the command records the date from the e-mail message as the commit author date, and uses the time of commit creation as the committer date. This allows the user to lie about the author date by using the same value as the committer date.

--skip

Skip the current patch. This is only meaningful when restarting an aborted patch.

-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**.

--continue, -r, --resolved
: After a patch failure (e.g. attempting to apply conflicting patch), the user has applied it by hand and the index file stores the result of the application. Make a commit using the authorship and commit log extracted from the e-mail message and the current index file, and continue.

--resolvemsg=<msg>
: When a patch failure occurs, <msg> will be printed to the screen before exiting. This overrides the standard message informing you to use **--continue** or **--skip** to handle the failure. This is solely for internal use between *git rebase* and *git am*.

--abort
: Restore the original branch and abort the patching operation. Revert contents of files involved in the am operation to their pre-am state.

--quit
: Abort the patching operation but keep HEAD and the index untouched.

--show-current-patch[=(diff|raw)]
: Show the message at which **git am** has stopped due to conflicts. If **raw** is specified, show the raw contents of the e-mail message; if **diff**, show the diff portion only. Defaults to **raw**.

--allow-empty
: After a patch failure on an input e-mail message lacking a patch, create an empty commit with the contents of the e-mail message as its log message.

DISCUSSION

The commit author name is taken from the "From: " line of the message, and commit author date is taken from the "Date: " line of the message. The "Subject: " line is used as the title of the commit, after stripping common prefix "[PATCH <anything>]". The "Subject: " line is supposed to concisely describe what the commit is about in one line of text.

"From: ", "Date: ", and "Subject: " lines starting the body override the respective commit author name and title values taken from the headers.

The commit message is formed by the title taken from the "Subject: ", a blank line and the body of the

message up to where the patch begins. Excess whitespace at the end of each line is automatically stripped.

The patch is expected to be inline, directly following the message. Any line that is of the form:

- and end-of-line, or

- line that begins with "diff -", or

- line that begins with "Index: "

is taken as the beginning of a patch, and the commit log message is terminated before the first occurrence of such a line.

When initially invoking **git am**, you give it the names of the mailboxes to process. Upon seeing the first patch that does not apply, it aborts in the middle. You can recover from this in one of two ways:

1. the current patch by re-running the command with the **--skip** option.

2. resolve the conflict in the working directory, and update the index file to bring it into a state that the patch should have produced. Then run the command with the **--continue** option.

The command refuses to process new mailboxes until the current operation is finished, so if you decide to start over from scratch, run **git am --abort** before running the command with mailbox names.

Before any patches are applied, ORIG_HEAD is set to the tip of the current branch. This is useful if you have problems with multiple commits, like running *git am* on the wrong branch or an error in the commits that is more easily fixed by changing the mailbox (e.g. errors in the "From:" lines).

HOOKS

This command can run **applypatch-msg**, **pre-applypatch**, and **post-applypatch** hooks. See **githooks**(5) for more information.

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:

am.keepcr
: If true, git-am will call git-mailsplit for patches in mbox format with parameter **--keep-cr**. In this case git-mailsplit will not remove **\r** from lines ending with **\r\n**. Can be overridden by giving **--no-keep-cr** from the command line. See **git-am**(1), **git-mailsplit**(1).

am.threeWay
: By default, **git am** will fail if the patch does not apply cleanly. When set to true, this setting tells **git am** to fall back on 3-way merge if the patch records the identity of blobs it is supposed to apply to and we have those blobs available locally (equivalent to giving the **--3way** option from the command line). Defaults to **false**. See **git-am**(1).

SEE ALSO

git-apply(1), **git-format-patch**(1).

GIT

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