NAME

   git-interpret-trailers − Add or parse structured information in commit messages

SYNOPSIS

   git interpret−trailers [<options>] [(−−trailer <token>[(=|:)<value>])...] [<file>...]
   git interpret−trailers [<options>] [−−parse] [<file>...]

DESCRIPTION

   Help parsing or adding *trailers* lines, that look similar to RFC 822 e−mail headers, at the end of the
   otherwise free−form part of a commit message.

   This command reads some patches or commit messages from either the <file> arguments or the standard
   input if no <file> is specified. If −−**parse** is specified, the output consists of the parsed trailers.

   Otherwise, this command applies the arguments passed using the −−**trailer** option, if any, to the commit
   message part of each input file. The result is emitted on the standard output.

   Some configuration variables control the way the −−**trailer** arguments are applied to each commit message
   and the way any existing trailer in the commit message is changed. They also make it possible to
   automatically add some trailers.

   By default, a *<token>=<value>* or *<token>:<value>* argument given using −−**trailer** will be appended
   after the existing trailers only if the last trailer has a different (<token>, <value>) pair (or if there is no
   existing trailer). The <token> and <value> parts will be trimmed to remove starting and trailing whitespace,
   and the resulting trimmed <token> and <value> will appear in the message like this:

   token: value

   This means that the trimmed <token> and <value> will be separated by '**:** ' (one colon followed by one
   space).

   By default the new trailer will appear at the end of all the existing trailers. If there is no existing trailer, the
   new trailer will appear after the commit message part of the output, and, if there is no line with only spaces
   at the end of the commit message part, one blank line will be added before the new trailer.

   Existing trailers are extracted from the input message by looking for a group of one or more lines that (i) is
   all trailers, or (ii) contains at least one Git−generated or user−configured trailer and consists of at least 25%
   trailers. The group must be preceded by one or more empty (or whitespace−only) lines. The group must
   either be at the end of the message or be the last non−whitespace lines before a line that starts with − − −
   (followed by a space or the end of the line). Such three minus signs start the patch part of the message. See
   also −−**no−divider** below.

   When reading trailers, there can be whitespaces after the token, the separator and the value. There can also
   be whitespaces inside the token and the value. The value may be split over multiple lines with each
   subsequent line starting with whitespace, like the "folding" in RFC 822.

   Note that *trailers* do not follow and are not intended to follow many rules for RFC 822 headers. For
   example they do not follow the encoding rules and probably many other rules.

OPTIONS

   −−in−place
      Edit the files in place.

   −−trim−empty
      If the <value> part of any trailer contains only whitespace, the whole trailer will be removed from the

resulting message. This applies to existing trailers as well as new trailers.

−−trailer <token>[(=|:)<value>]
  Specify a (<token>, <value>) pair that should be applied as a trailer to the input messages. See the description of this command.

−−where <placement>, −−no−where
  Specify where all new trailers will be added. A setting provided with −−*where* overrides all configuration variables and applies to all −−*trailer* options until the next occurrence of −−*where* or −−*no−where*. Possible values are **after**, **before**, **end** or **start**.

−−if−exists <action>, −−no−if−exists
  Specify what action will be performed when there is already at least one trailer with the same <token> in the message. A setting provided with −−*if−exists* overrides all configuration variables and applies to all −−*trailer* options until the next occurrence of −−*if−exists* or −−*no−if−exists*. Possible actions are **addIfDifferent**, **addIfDifferentNeighbor**, **add**, **replace** and **doNothing**.

−−if−missing <action>, −−no−if−missing
  Specify what action will be performed when there is no other trailer with the same <token> in the message. A setting provided with −−*if−missing* overrides all configuration variables and applies to all −−*trailer* options until the next occurrence of −−*if−missing* or −−*no−if−missing*. Possible actions are **doNothing** or **add**.

−−only−trailers
  Output only the trailers, not any other parts of the input.

−−only−input
  Output only trailers that exist in the input; do not add any from the command−line or by following configured **trailer.*** rules.

−−unfold
  Remove any whitespace−continuation in trailers, so that each trailer appears on a line by itself with its full content.

−−parse
  A convenience alias for **−−only−trailers −−only−input −−unfold**.

−−no−divider
  Do not treat −−− as the end of the commit message. Use this when you know your input contains just the commit message itself (and not an email or the output of **git format−patch**).

CONFIGURATION VARIABLES

trailer.separators
  This option tells which characters are recognized as trailer separators. By default only *:* is recognized as a trailer separator, except that = is always accepted on the command line for compatibility with other git commands.

  The first character given by this option will be the default character used when another separator is not specified in the config for this trailer.

  For example, if the value for this option is "%=$", then only lines using the format *<token><sep><value>* with <sep> containing *%*, *=* or *$* and then spaces will be considered trailers. And *%* will be the default separator used, so by default trailers will appear like: *<token>% <value>* (one percent sign and one space will appear between the token and the value).

trailer.where
  This option tells where a new trailer will be added.

  This can be **end**, which is the default, **start**, **after** or **before**.

  If it is **end**, then each new trailer will appear at the end of the existing trailers.

If it is **start**, then each new trailer will appear at the start, instead of the end, of the existing trailers.

If it is **after**, then each new trailer will appear just after the last trailer with the same <token>.

If it is **before**, then each new trailer will appear just before the first trailer with the same <token>.

trailer.ifexists
This option makes it possible to choose what action will be performed when there is already at least one trailer with the same <token> in the message.

The valid values for this option are: **addIfDifferentNeighbor** (this is the default), **addIfDifferent**, **add**, **replace** or **doNothing**.

With **addIfDifferentNeighbor**, a new trailer will be added only if no trailer with the same (<token>, <value>) pair is above or below the line where the new trailer will be added.

With **addIfDifferent**, a new trailer will be added only if no trailer with the same (<token>, <value>) pair is already in the message.

With **add**, a new trailer will be added, even if some trailers with the same (<token>, <value>) pair are already in the message.

With **replace**, an existing trailer with the same <token> will be deleted and the new trailer will be added. The deleted trailer will be the closest one (with the same <token>) to the place where the new one will be added.

With **doNothing**, nothing will be done; that is no new trailer will be added if there is already one with the same <token> in the message.

trailer.ifmissing
This option makes it possible to choose what action will be performed when there is not yet any trailer with the same <token> in the message.

The valid values for this option are: **add** (this is the default) and **doNothing**.

With **add**, a new trailer will be added.

With **doNothing**, nothing will be done.

trailer.<token>.key
This **key** will be used instead of <token> in the trailer. At the end of this key, a separator can appear and then some space characters. By default the only valid separator is *:*, but this can be changed using the **trailer.separators** config variable.

If there is a separator, then the key will be used instead of both the <token> and the default separator when adding the trailer.

trailer.<token>.where
This option takes the same values as the *trailer.where* configuration variable and it overrides what is specified by that option for trailers with the specified <token>.

trailer.<token>.ifexists
This option takes the same values as the *trailer.ifexists* configuration variable and it overrides what is specified by that option for trailers with the specified <token>.

trailer.<token>.ifmissing
This option takes the same values as the *trailer.ifmissing* configuration variable and it overrides what is specified by that option for trailers with the specified <token>.

trailer.<token>.command

>   This option behaves in the same way as *trailer.<token>.cmd*, except that it doesn't pass anything as argument to the specified command. Instead the first occurrence of substring $ARG is replaced by the value that would be passed as argument.
>
>   The *trailer.<token>.command* option has been deprecated in favor of *trailer.<token>.cmd* due to the fact that $ARG in the user's command is only replaced once and that the original way of replacing $ARG is not safe.
>
>   When both *trailer.<token>.cmd* and *trailer.<token>.command* are given for the same <token>, *trailer.<token>.cmd* is used and *trailer.<token>.command* is ignored.

trailer.<token>.cmd

>   This option can be used to specify a shell command that will be called: once to automatically add a trailer with the specified <token>, and then each time a *−−trailer <token>=<value>* argument to modify the <value> of the trailer that this option would produce.
>
>   When the specified command is first called to add a trailer with the specified <token>, the behavior is as if a special *−−trailer <token>=<value>* argument was added at the beginning of the "git interpret−trailers" command, where <value> is taken to be the standard output of the command with any leading and trailing whitespace trimmed off.
>
>   If some *−−trailer <token>=<value>* arguments are also passed on the command line, the command is called again once for each of these arguments with the same <token>. And the <value> part of these arguments, if any, will be passed to the command as its first argument. This way the command can produce a <value> computed from the <value> passed in the *−−trailer <token>=<value>* argument.

EXAMPLES

• Configure a *sign* trailer with a *Signed−off−by* key, and then add two of these trailers to a message:

```
$ git config trailer.sign.key "Signed−off−by"
$ cat msg.txt
subject

message
$ cat msg.txt | git interpret−trailers −−trailer 'sign: Alice <alice@example.com>' −−trailer 'sign: Bob <bob@exam
subject

message

Signed−off−by: Alice <alice@example.com>
Signed−off−by: Bob <bob@example.com>
```

• Use the **−−in−place** option to edit a message file in place:

```
$ cat msg.txt
subject

message

Signed−off−by: Bob <bob@example.com>
$ git interpret−trailers −−trailer 'Acked−by: Alice <alice@example.com>' −−in−place msg.txt
$ cat msg.txt
subject
```

message

Signed−off−by: Bob <bob@example.com>
Acked−by: Alice <alice@example.com>

- Extract the last commit as a patch, and add a *Cc* and a *Reviewed−by* trailer to it:

  $ git format−patch −1
  0001−foo.patch
  $ git interpret−trailers −−trailer 'Cc: Alice <alice@example.com>' −−trailer 'Reviewed−by: Bob <bob@example.c

- Configure a *sign* trailer with a command to automatically add a 'Signed−off−by: ' with the author information only if there is no 'Signed−off−by: ' already, and show how it works:

  $ git config trailer.sign.key "Signed−off−by: "
  $ git config trailer.sign.ifmissing add
  $ git config trailer.sign.ifexists doNothing
  $ git config trailer.sign.command 'echo "$(git config user.name) <$(git config user.email)>"'
  $ git interpret−trailers <<EOF
  > EOF

  Signed−off−by: Bob <bob@example.com>
  $ git interpret−trailers <<EOF
  > Signed−off−by: Alice <alice@example.com>
  > EOF

  Signed−off−by: Alice <alice@example.com>

- Configure a *fix* trailer with a key that contains a # and no space after this character, and show how it works:

  $ git config trailer.separators ":#"
  $ git config trailer.fix.key "Fix #"
  $ echo "subject" | git interpret−trailers −−trailer fix=42
  subject

  Fix #42

- Configure a *help* trailer with a cmd use a script **glog−find−author** which search specified author identity from git log in git repository and show how it works:

  $ cat ˜/bin/glog−find−author
  #!/bin/sh
  test −n "$1" && git log −−author="$1" −−pretty="%an <%ae>" −1 || true
  $ git config trailer.help.key "Helped−by: "
  $ git config trailer.help.ifExists "addIfDifferentNeighbor"
  $ git config trailer.help.cmd "˜/bin/glog−find−author"
  $ git interpret−trailers −−trailer="help:Junio" −−trailer="help:Couder" <<EOF
  > subject
  >
  > message
  >
  > EOF

subject

message

Helped−by: Junio C Hamano <gitster@pobox.com>
Helped−by: Christian Couder <christian.couder@gmail.com>

- Configure a *ref* trailer with a cmd use a script **glog−grep** to grep last relevant commit from git log
  in the git repository and show how it works:

  ```
  $ cat ˜/bin/glog−grep
  #!/bin/sh
  test −n "$1" && git log −−grep "$1" −−pretty=reference −1 || true
  $ git config trailer.ref.key "Reference−to: "
  $ git config trailer.ref.ifExists "replace"
  $ git config trailer.ref.cmd "˜/bin/glog−grep"
  $ git interpret−trailers −−trailer="ref:Add copyright notices." <<EOF
  > subject
  >
  > message
  >
  > EOF
  subject

  message

  Reference−to: 8bc9a0c769 (Add copyright notices., 2005−04−07)
  ```

- Configure a *see* trailer with a command to show the subject of a commit that is related, and show
  how it works:

  ```
  $ git config trailer.see.key "See−also: "
  $ git config trailer.see.ifExists "replace"
  $ git config trailer.see.ifMissing "doNothing"
  $ git config trailer.see.command "git log −1 −−oneline −−format=\"%h (%s)\" −−abbrev−commit −−abbrev=14 \$
  $ git interpret−trailers <<EOF
  > subject
  >
  > message
  >
  > see: HEAD˜2
  > EOF
  subject

  message

  See−also: fe3187489d69c4 (subject of related commit)
  ```

- Configure a commit template with some trailers with empty values (using sed to show and keep the
  trailing spaces at the end of the trailers), then configure a commit−msg hook that uses *git
  interpret−trailers* to remove trailers with empty values and to add a *git−version* trailer:

  ```
  $ sed −e 's/ Z$/ /' >commit_template.txt <<EOF
  ```

```
> ***subject***
>
> ***message***
>
> Fixes: Z
> Cc: Z
> Reviewed−by: Z
> Signed−off−by: Z
> EOF
$ git config commit.template commit_template.txt
$ cat >.git/hooks/commit−msg <<EOF
> #!/bin/sh
> git interpret−trailers −−trim−empty −−trailer "git−version: \$(git describe)" "\$1" > "\$1.new"
> mv "\$1.new" "\$1"
> EOF
$ chmod +x .git/hooks/commit−msg
```

SEE ALSO

git-commit(1), **git-format-patch**(1), **git-config**(1)

GIT

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