Rocky Enterprise Linux 9.2 Manual Pages on command 'git-mailinfo.1'

$ man git-mailinfo.1

```
GIT-MAILINFO(1)                    Git Manual                    GIT-MAILINFO(1)

NAME

       git-mailinfo - Extracts patch and authorship from a single e-mail message

SYNOPSIS

       git mailinfo [-k|-b] [-u | --encoding=<encoding> | -n]

                     [--[no-]scissors] [--quoted-cr=<action>]

                     <msg> <patch>

DESCRIPTION

       Reads a single e-mail message from the standard input, and writes the commit log message

       in <msg> file, and the patches in <patch> file. The author name, e-mail and e-mail subject

       are written out to the standard output to be used by git am to create a commit. It is

       usually not necessary to use this command directly. See git-am(1) instead.

OPTIONS

       -k

           Usually the program removes email cruft from the Subject: header line to extract the

           title line for the commit log message. This option prevents this munging, and is most

           useful when used to read back git format-patch -k output.

           Specifically, the following are removed until none of them remain:

           ?   Leading and trailing whitespace.

           ?   Leading Re:, re:, and :.

           ?   Leading bracketed strings (between [ and ], usually [PATCH]).

           Finally, runs of whitespace are normalized to a single ASCII space character.

       -b
```

When -k is not in effect, all leading strings bracketed with [and] pairs are stripped. This option limits the stripping to only the pairs whose bracketed string contains the word "PATCH".

-u

The commit log message, author name and author email are taken from the e-mail, and after minimally decoding MIME transfer encoding, re-coded in the charset specified by i18n.commitEncoding (defaulting to UTF-8) by transliterating them. This used to be optional but now it is the default.

Note that the patch is always used as-is without charset conversion, even with this flag.

--encoding=<encoding>

Similar to -u. But when re-coding, the charset specified here is used instead of the one specified by i18n.commitEncoding or UTF-8.

-n

Disable all charset re-coding of the metadata.

-m, --message-id

Copy the Message-ID header at the end of the commit message. This is useful in order to associate commits with mailing list discussions.

--scissors

Remove everything in body before a scissors line (e.g. "-- >8 --"). The line represents scissors and perforation marks, and is used to request the reader to cut the message at that line. If that line appears in the body of the message before the patch, everything before it (including the scissors line itself) is ignored when this option is used.

This is useful if you want to begin your message in a discussion thread with comments and suggestions on the message you are responding to, and to conclude it with a patch submission, separating the discussion and the beginning of the proposed commit log message with a scissors line.

This can be enabled by default with the configuration option mailinfo.scissors.

--no-scissors

Ignore scissors lines. Useful for overriding mailinfo.scissors settings.

--quoted-cr=<action>

Action when processes email messages sent with base64 or quoted-printable encoding, and the decoded lines end with a CRLF instead of a simple LF.

The valid actions are:

? nowarn: Git will do nothing when such a CRLF is found.

? warn: Git will issue a warning for each message if such a CRLF is found.

? strip: Git will convert those CRLF to LF.

The default action could be set by configuration option mailinfo.quotedCR. If no such configuration option has been set, warn will be used.

<msg>

The commit log message extracted from e-mail, usually except the title line which comes from e-mail Subject.

<patch>

The patch extracted from e-mail.

GIT

Part of the git(1) suite

Git 2.34.1 07/07/2023 GIT-MAILINFO(1)