NAME

gitmailmap - Map author/committer names and/or E-Mail addresses

SYNOPSIS

$GIT_WORK_TREE/.mailmap

DESCRIPTION

If the file **.mailmap** exists at the toplevel of the repository, or at the location pointed to by the **mailmap.file** or **mailmap.blob** configuration options (see **git-config**(1)), it is used to map author and committer names and email addresses to canonical real names and email addresses.

SYNTAX

The # character begins a comment to the end of line, blank lines are ignored.

In the simple form, each line in the file consists of the canonical real name of an author, whitespace, and an email address used in the commit (enclosed by < and >) to map to the name. For example:

Proper Name <commit@email.xx>

The more complex forms are:

<proper@email.xx> <commit@email.xx>

which allows mailmap to replace only the email part of a commit, and:

Proper Name <proper@email.xx> <commit@email.xx>

which allows mailmap to replace both the name and the email of a commit matching the specified commit email address, and:

Proper Name <proper@email.xx> Commit Name <commit@email.xx>

which allows mailmap to replace both the name and the email of a commit matching both the specified commit name and email address.

Both E-Mails and names are matched case-insensitively. For example this would also match the *Commit Name* <*commit@email.xx*> above:

Proper Name <proper@email.xx> CoMmIt NaMe <CoMmIt@EmAiL.xX>

NOTES

Git does not follow symbolic links when accessing a **.mailmap** file in the working tree. This keeps behavior consistent when the file is accessed from the index or a tree versus from the filesystem.

EXAMPLES

Your history contains commits by two authors, Jane and Joe, whose names appear in the repository under several forms:

```
Joe Developer <joe@example.com>
Joe R. Developer <joe@example.com>
Jane Doe <jane@example.com>
Jane Doe <jane@laptop.(none)>
Jane D. <jane@desktop.(none)>
```

Now suppose that Joe wants his middle name initial used, and Jane prefers her family name fully spelled out. A **.mailmap** file to correct the names would look like:

```
Joe R. Developer <joe@example.com>
Jane Doe <jane@example.com>
Jane Doe <jane@desktop.(none)>
```

Note that there's no need to map the name for <*jane@laptop.(none)*> to only correct the names. However, leaving the obviously broken <*jane@laptop.(none)*> and <*jane@desktop.(none)*> E-Mails as-is is usually not what you want. A **.mailmap** file which also corrects those is:

```
Joe R. Developer <joe@example.com>
Jane Doe <jane@example.com> <jane@laptop.(none)>
Jane Doe <jane@example.com> <jane@desktop.(none)>
```

Finally, let's say that Joe and Jane shared an E-Mail address, but not a name, e.g. by having these two commits in the history generated by a bug reporting system. I.e. names appearing in history as:

```
Joe <bugs@example.com>
Jane <bugs@example.com>
```

A full **.mailmap** file which also handles those cases (an addition of two lines to the above example)

would be:

```
Joe R. Developer <joe@example.com>
Jane Doe <jane@example.com> <jane@laptop.(none)>
Jane Doe <jane@example.com> <jane@desktop.(none)>
Joe R. Developer <joe@example.com> Joe <bugs@example.com>
Jane Doe <jane@example.com> Jane <bugs@example.com>
```

SEE ALSO

git-check-mailmap(1)

GIT

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