Creating a commit with multiple authors

In this article

You can attribute a commit to more than one author by adding one or more `Co-authored-by` trailers to the commit's message. Co-authored commits are visible on GitHub Enterprise Cloud.

Required co-author information

Before you can add a co-author to a commit, you must know the appropriate email to use for each co-author. For the co-author's commit to count as a contribution, you must use the email associated with their account on GitHub.com.

If a person chooses to keep their email address private, you should use their GitHub Enterprise Cloud-provided `no-reply` email to protect their privacy. Otherwise, the co-author's email will be available to the public in the commit message. If you want to keep your email private, you can choose to use a GitHub Enterprise Cloud-provided `no-reply` email for Git operations and ask other co-authors to list your `no-reply` email in commit trailers.

For more information, see "Setting your commit email address."

Tip: You can help a co-author find their preferred email address by sharing this information:

- To find your GitHub Enterprise Cloud-provided `no-reply` email, navigate to your email settings page under "Keep my email address private."
- To find the email you used to configure Git on your computer, run `git config user.email` on the command line.

Creating co-authored commits using GitHub Desktop

You can use GitHub Desktop to create a commit with a co-author. For more information, see "Committing and reviewing changes to your project in GitHub Desktop" and GitHub Desktop.

Creating co-authored commits on the command line

1. Collect the name and email address for each co-author. If a person chooses to keep their email address private, you should use their GitHub Enterprise Cloud-provided `no-reply` email to protect their privacy.
2. Type your commit message and a short, meaningful description of your changes. After your commit description, instead of a closing quotation, add two empty lines.

   ```
   $ git commit -m "Refactor usability tests.
   >
   >
   ```

   Tip: If you're using a text editor on the command line to type your commit message, ensure there are two newlines between the end of your commit description and the `Co-authored-by:` commit trailer.

3. On the next line of the commit message, type `Co-authored-by: name <name@example.com>` with specific information for each co-author. After the co-author information, add a closing quotation mark.

   If you're adding multiple co-authors, give each co-author their own line and `Co-authored-by:` commit trailer. Do not add blank lines between each co-author line.

   ```
   $ git commit -m "Refactor usability tests.
   >
   >
   Co-authored-by: NAME <NAME@EXAMPLE.COM>
   Co-authored-by: ANOTHER-NAME <ANOTHER-NAME@EXAMPLE.COM>"
   ```

The new commit and message will appear on GitHub.com the next time you push. For more information, see "Pushing commits to a remote repository."

Creating co-authored commits on GitHub Enterprise Cloud

After you've made changes in a file using the web editor on GitHub Enterprise Cloud, you can create a co-authored commit by adding a `Co-authored-by:` trailer to the commit's message.

1. Collect the name and email address for each co-author. If a person chooses to keep their email address private, you should use their GitHub Enterprise Cloud-provided `no-reply` email to protect their privacy.
2. Click **Commit changes...**
3. In the "Commit message" field, type a short, meaningful commit message that describes the changes you made.
4. In the text box below your commit message, add `Co-authored-by: name <name@example.com>` with specific information for each co-author. If you're adding multiple co-authors, give each co-author their own line and `Co-authored-by:` commit trailer.
5. Click **Commit changes** or **Propose changes**.

The new commit and message will appear on GitHub.com.

Further reading

- "Viewing a project's contributors"
- "Changing a commit message"
- "Committing and reviewing changes to your project in GitHub Desktop" in the GitHub Desktop documentation