

Viewing a file

In this article

Viewing or copying the raw file content

Viewing the line-by-line revision history for a file

Ignore commits in the blame view

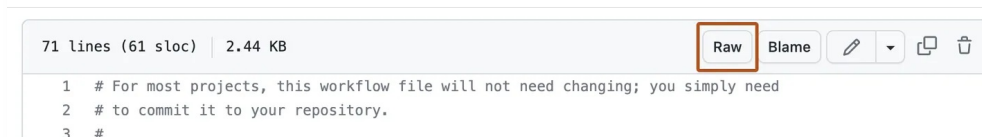
Bypassing .git-blame-ignore-revs in the blame view


You can view raw file content or trace changes to lines in a file and discover how parts of the file evolved over time.

Viewing or copying the raw file content

With the raw view, you can view or copy the raw content of a file without any styling.

- 1 On your GitHub Enterprise Server instance, navigate to the main page of the repository.
- 2 Click the file that you want to view.
- 3 In the upper-right corner of the file view, click **Raw**.



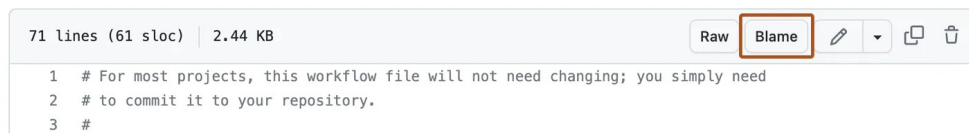
- 4 Optionally, to copy the raw file content, in the upper-right corner of the file view, click .


Viewing the line-by-line revision history for a file

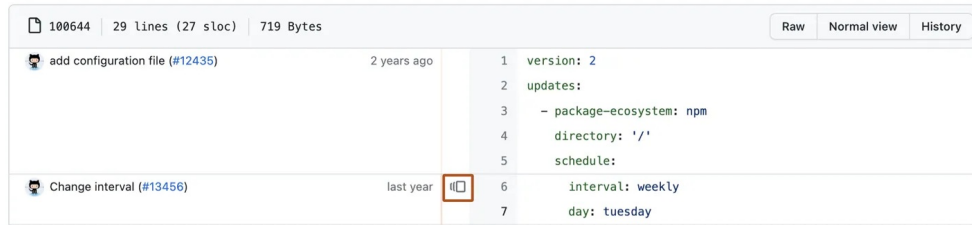
Within the blame view, you can view the line-by-line revision history for an entire file.

Tip: On the command line, you can also use `git blame` to view the revision history of lines within a file. For more information, see [Git's git blame documentation](#).

- 1 On your GitHub Enterprise Server instance, navigate to the main page of the repository.
- 2 Click to open the file whose line history you want to view.
- 3 In the upper-right corner of the file view, click **Blame** to open the blame view.



- 4 To see earlier revisions of a specific line, or reblame, click  until you've found the changes you're interested in viewing.



Ignore commits in the blame view

All revisions specified in the `.git-blame-ignore-revs` file, which must be in the root directory of your repository, are hidden from the blame view using Git's `git blame --ignore-revs-file` configuration setting. For more information, see [git blame --ignore-revs-file](#) in the Git documentation.

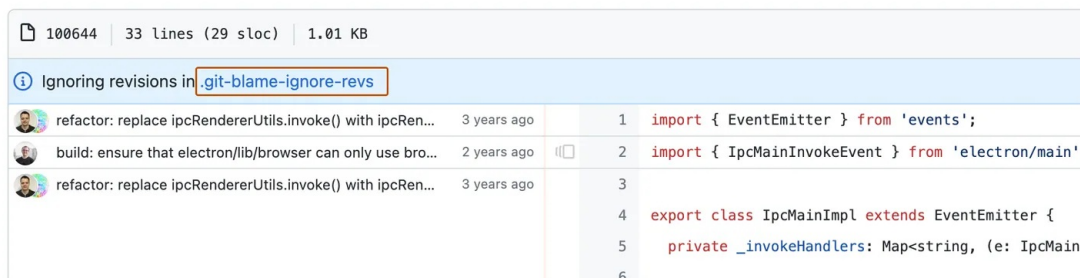
- 1 In the root directory of your repository, create a file named `.git-blame-ignore-revs`.
- 2 Add the commit hashes you want to exclude from the blame view to that file. We recommend the file to be structured as follows, including comments:

```
# .git-blame-ignore-revs
# Removed semi-colons from the entire codebase
a8940f7fbddf7fad9d7d50014d4e8d46baf30592
# Converted all JavaScript to TypeScript
69d029cec8337c616552756310748c4a507bd75a
```

- 3 Commit and push the changes.

Now when you visit the blame view, the listed revisions will not be included in the blame. You'll see an **Ignoring revisions in .git-blame-ignore-revs** banner indicating that some commits may be hidden:

[electron](#) / [lib](#) / [browser](#) / [ipc-main-impl.ts](#) 



This can be useful when a few commits make extensive changes to your code. You can use the file when running `git blame` locally as well:

```
git blame --ignore-revs-file .git-blame-ignore-revs
```

You can also configure your local git so it always ignores the revs in that file:

```
git config blame.ignoreRevsFile .git-blame-ignore-revs
```

Bypassing `.git-blame-ignore-revs` in the blame view [↗](#)

If the blame view for a file shows **Ignoring revisions in `.git-blame-ignore-revs`**, you can still bypass `.git-blame-ignore-revs` and see the normal blame view. In the URL, append a `~` to the SHA and the **Ignoring revisions in `.git-blame-ignore-revs`** banner will disappear.

Legal

© 2023 GitHub, Inc. [Terms](#) [Privacy](#) [Status](#) [Pricing](#) [Expert services](#) [Blog](#)