About code owners

In this article

You can use a CODEOWNERS file to define individuals or teams that are responsible for code in a repository.

> **Who can use this feature**
> People with write permissions for the repository can create or edit the CODEOWNERS file and be listed as code owners. People with admin or owner permissions can require that pull requests have to be approved by code owners before they can be merged.

You can define code owners in public repositories with GitHub Free and GitHub Free for organizations, and in public and private repositories with GitHub Pro, GitHub Team, GitHub Enterprise Cloud, and GitHub Enterprise Server. For more information, see "GitHub's plans."

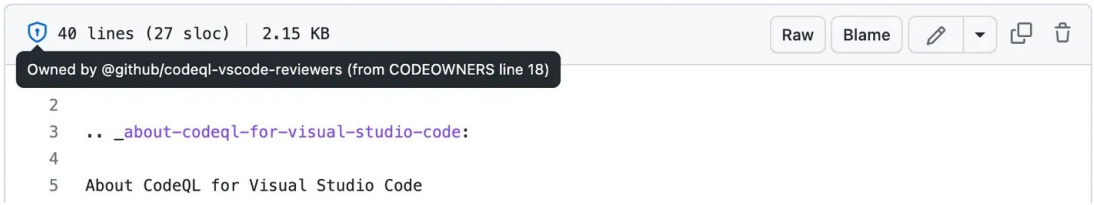
The people you choose as code owners must have write permissions for the repository. When the code owner is a team, that team must be visible and it must have write permissions, even if all the individual members of the team already have write permissions directly, through organization membership, or through another team membership.

About code owners

Code owners are automatically requested for review when someone opens a pull request that modifies code that they own. Code owners are not automatically requested to review draft pull requests. For more information about draft pull requests, see "About pull requests." When you mark a draft pull request as ready for review, code owners are automatically notified. If you convert a pull request to a draft, people who are already subscribed to notifications are not automatically unsubscribed. For more information, see "Changing the stage of a pull request."

When someone with admin or owner permissions has enabled required reviews, they also can optionally require approval from a code owner before the author can merge a pull request in the repository. For more information, see "About protected branches."

If a file has a code owner, you can see who the code owner is before you open a pull request. In the repository, you can browse to the file and hover over to see a tool tip with codeownership details.



CODEOWNERS file location

To use a CODEOWNERS file, create a new file called `CODEOWNERS` in the `.github/`, root, or `docs/` directory of the repository, in the branch where you'd like to add the code owners. If `CODEOWNERS` files exist in more than one of those locations, GitHub will search for them in that order and use the first one it finds.

Each CODEOWNERS file assigns the code owners for a single branch in the repository. Thus, you can assign different code owners for different branches, such as `@octo-org/codeowners-team` for a code base on the default branch and `@octocat` for a GitHub Pages site on the `gh-pages` branch.

For code owners to receive review requests, the CODEOWNERS file must be on the base branch of the pull request. For example, if you assign `@octocat` as the code owner for *.js* files on the `gh-pages` branch of your repository, `@octocat` will receive review requests when a pull request with changes to *.js* files is opened between the head branch and `gh-pages`.

CODEOWNERS and forks

To trigger review requests, pull requests use the version of `CODEOWNERS` from the base branch of the pull request. The base branch is the branch that a pull request will modify if the pull request is merged.

If you create a pull request from a fork, and the base branch is in the upstream repository, then the pull request will use the `CODEOWNERS` file from that branch in the upstream repository. If the base branch is a branch within your fork, then the pull request will use the `CODEOWNERS` file from that branch in your fork, but this will only trigger review requests if the code owners are added to your fork specifically with `write` access.

When you view who is responsible for a file by hovering over , you will see information from the `CODEOWNERS` file for whichever branch in whichever repository you're looking at.

CODEOWNERS file size

CODEOWNERS files must be under 3 MB in size. A CODEOWNERS file over this limit will not be loaded, which means that code owner information is not shown and the appropriate code owners will not be requested to review changes in a pull request.

To reduce the size of your CODEOWNERS file, consider using wildcard patterns to consolidate multiple entries into a single entry.

CODEOWNERS syntax

Warning: There are some syntax rules for gitignore files that *do not work* in CODEOWNERS files:

- Escaping a pattern starting with `#` using `\` so it is treated as a pattern and not a comment
- Using `!` to negate a pattern
- Using `[ ]` to define a character range

A CODEOWNERS file uses a pattern that follows most of the same rules used in gitignore files. The pattern is followed by one or more GitHub usernames or team names using the standard `@username` or `@org/team-name` format. Users and teams must have explicit `write` access to the repository, even if the team's members already have access.

If you want to match two or more code owners with the same pattern, all the code owners must be on the same line. If the code owners are not on the same line, the pattern matches only the last mentioned code owner.

In most cases, you can also refer to a user by an email address that has been added to their account on GitHub.com, for example `user@example.com`. You cannot use an email address to refer to a managed user account. For more information about managed user accounts, see "About Enterprise Managed Users."

CODEOWNERS paths are case sensitive, because GitHub uses a case sensitive file system. Since CODEOWNERS are evaluated by GitHub, even systems that are case insensitive (for example, macOS) must use paths and files that are cased correctly in the CODEOWNERS file.

If any line in your CODEOWNERS file contains invalid syntax, that line will be skipped. When you navigate to the CODEOWNERS file in your repository on GitHub.com, you can see any errors highlighted. A list of errors in a repository's CODEOWNERS file is also accessible via the API. For more information, see "Repositories" in the REST API documentation.

Example of a CODEOWNERS file

```
# This is a comment.
# Each line is a file pattern followed by one or more owners.

# These owners will be the default owners for everything in
# the repo. Unless a later match takes precedence,
# @global-owner1 and @global-owner2 will be requested for
# review when someone opens a pull request.
*       @global-owner1 @global-owner2

# Order is important; the last matching pattern takes the most
# precedence. When someone opens a pull request that only
# modifies JS files, only @js-owner and not the global
# owner(s) will be requested for a review.
*.js    @js-owner #This is an inline comment.

# You can also use email addresses if you prefer. They'll be
# used to look up users just like we do for commit author
# emails.
*.go docs@example.com

# Teams can be specified as code owners as well. Teams should
# be identified in the format @org/team-name. Teams must have
# explicit write access to the repository. In this example,
# the octocats team in the octo-org organization owns all .txt files.
*.txt @octo-org/octocats

# In this example, @doctocat owns any files in the build/logs
# directory at the root of the repository and any of its
# subdirectories.
/build/logs/ @doctocat

# The `docs/*` pattern will match files like
# `docs/getting-started.md` but not further nested files like
# `docs/build-app/troubleshooting.md`.
docs/*  docs@example.com

# In this example, @octocat owns any file in an apps directory
# anywhere in your repository.
```

```
apps/ @octocat

# In this example, @doctocat owns any file in the `/docs`
# directory in the root of your repository and any of its
# subdirectories.
/docs/ @doctocat

# In this example, any change inside the `/scripts` directory
# will require approval from @doctocat or @octocat.
/scripts/ @doctocat @octocat

# In this example, @octocat owns any file in a `/logs` directory such as
# `/build/logs`, `/scripts/logs`, and `/deeply/nested/logs`. Any changes
# in a `/logs` directory will require approval from @octocat.
**/logs @octocat

# In this example, @octocat owns any file in the `/apps`
# directory in the root of your repository except for the `/apps/github`
# subdirectory, as its owners are left empty.
/apps/ @octocat
/apps/github

# In this example, @octocat owns any file in the `/apps`
# directory in the root of your repository except for the `/apps/github`
# subdirectory, as this subdirectory has its own owner @doctocat
/apps/ @octocat
/apps/github @doctocat
```

CODEOWNERS and branch protection

Repository owners can add branch protection rules to ensure that changed code is reviewed by the owners of the changed files. For more information, see "About protected branches."

Further reading

- "Creating new files"
- "Inviting collaborators to a personal repository"
- "Managing an individual's access to an organization repository"
- "Managing team access to an organization repository"
- "Viewing a pull request review"