Home

Pull requests

Commit changes to your project
Collaborate with pull requests
Getting started
Collaborative development
Best practices
Working with forks
Code quality features
Propose changes
Address merge conflicts
Review changes
Incorporate changes

Best practices for pull requests

You can follow best practices to improve the consistency and quality of pull requests and pull request reviews.

Best practices for creating pull requests

When creating a pull request, follow a few best practices for a smoother review process. For information on creating a pull request, see "Creating a pull request."

Write small PRs

Aim to create small, focused pull requests that fulfill a single purpose. Smaller pull requests are easier and faster to review and merge, leave less room to introduce bugs, and provide a clearer history of changes.

Review your own pull request first

Review, build, and test your own pull request before submitting it. This will allow you to catch errors or typos that you may have missed, before others start reviewing.

Provide context and guidance

Write clear titles and descriptions for your pull requests so that reviewers can quickly understand what the pull request does. In the pull request body, include:

- the purpose of the pull request
- an overview of what changed
- links to any additional context such as tracking issues or previous conversations

To help reviewers, share the type of feedback you need. For example, do you need a quick look or a deeper critique?

If your pull request consists of changes to multiple files, provide guidance to reviewers about the order in which to review the files. Recommend where to start and how to proceed with the review.

Best practices for managing pull requests

If you are a repository maintainer, take these steps to manage and standardize the pull requests that contributors create in your repository.

Use pull request templates

Pull request templates let you customize and standardize the information you'd like to be included when someone creates a pull request in your repository. When you add a pull request template to your repository, project contributors will automatically see the template's contents in the pull request body. For more information, see "Creating a pull request template for your repository."

You can use pull request templates to standardize the review process for your repository. For example, you can include a list of tasks that you would like authors to complete before merging their pull requests, by adding a task list to the template. For more information, see "About task lists."

You can request that contributors include an issue reference in their pull request body, so that merging the pull request will automatically close the issue. For more information, see "Linking a pull request to an issue."

Define code owners

You may want to make sure that specific individuals always review changes to certain code or files in your repository. For example, you may want a technical writer on your team to always review changes in the `docs` directory.

You can define individuals or teams that you consider responsible for code or files in a repository to be code owners. Code owners will automatically be requested for review when someone opens a pull request that modifies the files that they own. You can define code owners for specific types of files or directories, as well as for different branches in a repository. For more information, see "About code owners."

Use protected branches

You can use protected branches to prevent pull requests from being merged into important branches, such as `main`, until certain conditions are met. For example, you can require passing CI tests or an approving review. For more information, see "About protected branches."

Use automated tools to review code styling

Use automated tools, such as linters, in your repository's pull requests to maintain consistent styling and make code more understandable. Using automated tools to catch smaller problems like typos or styling leaves more time for reviewers to focus on the substance of a pull request.

For example, you can use GitHub Actions to set up code linters that can run on pull requests as part of your continuous integration (CI) workflow. For more information, see "About continuous integration."

Help and support

Did this doc help you?

Privacy policy

Help us make these docs great!

All GitHub docs are open source. See something that's wrong or unclear? Submit a pull request.

Make a contribution

Learn how to contribute

Still need help?

Ask the GitHub community

Contact support