# Canceling a workflow

In this article

You can cancel a workflow run that is in progress. When you cancel a workflow run, GitHub cancels all jobs and steps that are a part of that workflow.

> **Note:** GitHub-hosted runners are not currently supported on GitHub Enterprise Server. You can see more information about planned future support on the [GitHub public roadmap](#).
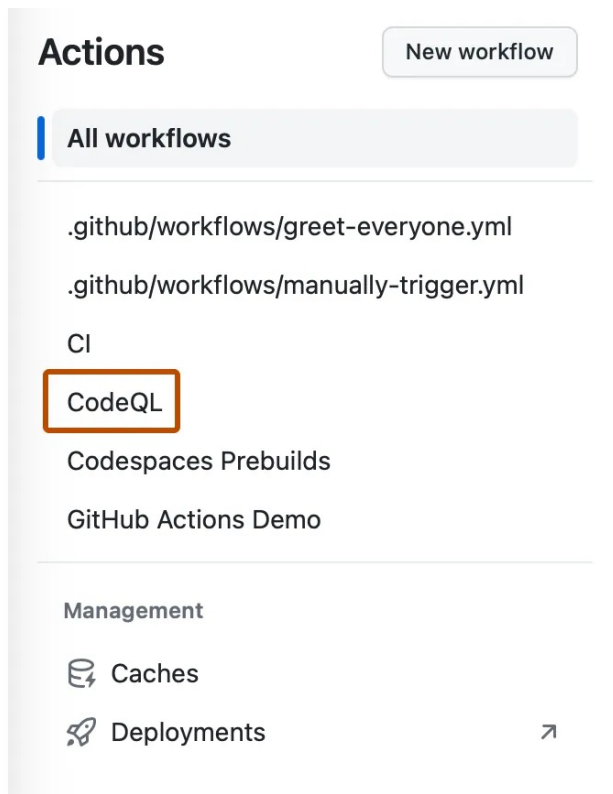
Write access to the repository is required to perform these steps.
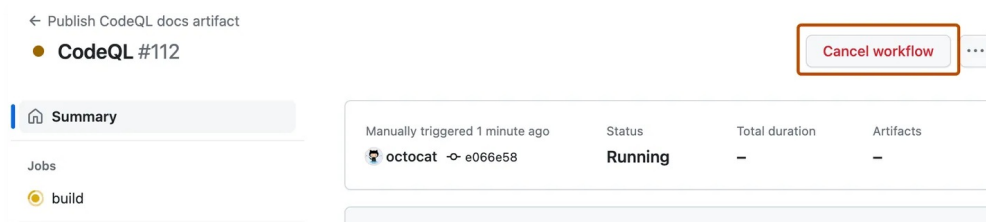
Canceling a workflow run 🔗

1. On your GitHub Enterprise Server instance, navigate to the main page of the repository.

2. Under your repository name, click ▶ **Actions**.
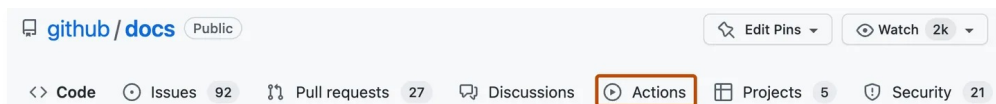
3. In the left sidebar, click the workflow you want to see.

④ From the list of workflow runs, click the name of the `queued` or `in progress` run that you want to cancel.

⑤ In the upper-right corner of the workflow, click **Cancel workflow**.

Steps GitHub takes to cancel a workflow run 🔗

When canceling workflow run, you may be running other software that uses resources that are related to the workflow run. To help you free up resources related to the workflow run, it may help to understand the steps GitHub performs to cancel a workflow run.

① To cancel the workflow run, the server re-evaluates `if` conditions for all currently running jobs. If the condition evaluates to `true`, the job will not get canceled. For example, the condition `if: always()` would evaluate to true and the job continues to run. When there is no condition, that is the equivalent of the condition `if: success()`, which only runs if the previous step finished successfully.

② For jobs that need to be canceled, the server sends a cancellation message to all the runner machines with jobs that need to be canceled.

③ For jobs that continue to run, the server re-evaluates `if` conditions for the unfinished steps. If the condition evaluates to `true`, the step continues to run.

④ For steps that need to be canceled, the runner machine sends `SIGINT/Ctrl-C` to the step's entry process (`node` for javascript action, `docker` for container action, and

`bash/cmd/pwd` when using `run` in a step). If the process doesn't exit within 7500 ms, the runner will send `SIGTERM/Ctrl-Break` to the process, then wait for 2500 ms for the process to exit. If the process is still running, the runner kills the process tree.

5. After the 5 minutes cancellation timeout period, the server will force terminate all jobs and steps that don't finish running or fail to complete the cancellation process.