Using concurrency

In this article

Run a single job at a time.

Overview 🔗

You can use `jobs.<job_id>.concurrency` to ensure that only a single job or workflow using the same concurrency group will run at a time. A concurrency group can be any string or expression. Allowed expression contexts: `github` , `inputs` , `vars` , `needs` , `strategy` , and `matrix` . For more information about expressions, see "Expressions."

You can also specify `concurrency` at the workflow level. For more information, see `concurrency` .

When a concurrent job or workflow is queued, if another job or workflow using the same concurrency group in the repository is in progress, the queued job or workflow will be `pending` . Any previously pending job or workflow in the concurrency group will be canceled. To also cancel any currently running job or workflow in the same concurrency group, specify `cancel-in-progress: true` .

> **Notes:**
>
> - The concurrency group name is case insensitive. For example, `prod` and `Prod` will be treated as the same concurrency group.
> - Ordering is not guaranteed for jobs or runs using concurrency groups, they are handled in the order that they are processed.

Examples: Using concurrency and the default behavior 🔗

```
concurrency: staging_environment
```

```
concurrency: ci-${{ github.ref }}
```

Example: Using concurrency to cancel any in-progress job or run 🔗

```
concurrency:
  group: ${{ github.ref }}
  cancel-in-progress: true
```

Example: Using a fallback value 🔗

If you build the group name with a property that is only defined for specific events, you can use a fallback value. For example, `github.head_ref` is only defined on `pull_request` events. If your workflow responds to other events in addition to `pull_request` events, you will need to provide a fallback to avoid a syntax error. The following concurrency group cancels in-progress jobs or runs on `pull_request` events only; if `github.head_ref` is undefined, the concurrency group will fallback to the run ID, which is guaranteed to be both unique and defined for the run.

```
concurrency:
  group: ${{ github.head_ref || github.run_id }}
  cancel-in-progress: true
```

Example: Only cancel in-progress jobs or runs for the current workflow 🔗

If you have multiple workflows in the same repository, concurrency group names must be unique across workflows to avoid canceling in-progress jobs or runs from other workflows. Otherwise, any previously in-progress or pending job will be canceled, regardless of the workflow.

To only cancel in-progress runs of the same workflow, you can use the `github.workflow` property to build the concurrency group:

```
concurrency:
  group: ${{ github.workflow }}-${{ github.ref }}
  cancel-in-progress: true
```

Monitoring your current jobs in your organization or enterprise 🔗

To identify any constraints with concurrency or queuing, you can check how many jobs are currently being processed on the GitHub-hosted runners in your organization or enterprise. For more information, see "Monitoring your current jobs."