

Assigning permissions to jobs

In this article

- Overview
- Defining access for the GITHUB_TOKEN scopes
- Setting the GITHUB_TOKEN permissions for all jobs in a workflow
- Setting the GITHUB_TOKEN permissions for a specific job

Modify the default permissions granted to GITHUB_TOKEN.

Overview [↗](#)

You can use permissions to modify the default permissions granted to the GITHUB_TOKEN, adding or removing access as required, so that you only allow the minimum required access. For more information, see ["Automatic token authentication."](#)

You can use permissions either as a top-level key, to apply to all jobs in the workflow, or within specific jobs. When you add the permissions key within a specific job, all actions and run commands within that job that use the GITHUB_TOKEN gain the access rights you specify. For more information, see [jobs.<job_id>.permissions](#).

For each of the available scopes, shown in the table below, you can assign one of the permissions: read, write, or none. If you specify the access for any of these scopes, all of those that are not specified are set to none.

Available scopes and details of what each allows an action to do:

Scope	Allows an action using GITHUB_TOKEN to
actions	Work with GitHub Actions. For example, actions: write permits an action to cancel a workflow run. For more information, see "Permissions required for GitHub Apps."
checks	Work with check runs and check suites. For example, checks: write permits an action to create a check run. For more information, see "Permissions required for GitHub Apps."
contents	Work with the contents of the repository. For example, contents: read permits an action to list the commits, and contents:write allows the action to create a release. For more information, see "Permissions required for GitHub Apps."
deployments	Work with deployments. For example, deployments: write permits an action to create a new deployment. For more information, see "Permissions required for GitHub Apps."
discussions	Work with GitHub Discussions. For example, discussions: write permits an action to close or delete a discussion. For more information, see

"[Using the GraphQL API for Discussions.](#)"

id-token	Fetch an OpenID Connect (OIDC) token. This requires <code>id-token: write</code> . For more information, see " About security hardening with OpenID Connect "
issues	Work with issues. For example, <code>issues: write</code> permits an action to add a comment to an issue. For more information, see " Permissions required for GitHub Apps. "
packages	Work with GitHub Packages. For example, <code>packages: write</code> permits an action to upload and publish packages on GitHub Packages. For more information, see " About permissions for GitHub Packages. "
pages	Work with GitHub Pages. For example, <code>pages: write</code> permits an action to request a GitHub Pages build. For more information, see " Permissions required for GitHub Apps. "
pull-requests	Work with pull requests. For example, <code>pull-requests: write</code> permits an action to add a label to a pull request. For more information, see " Permissions required for GitHub Apps. "
repository-projects	Work with GitHub projects (classic). For example, <code>repository-projects: write</code> permits an action to add a column to a project (classic). For more information, see " Permissions required for GitHub Apps. "
security-events	Work with GitHub code scanning and Dependabot alerts. For example, <code>security-events: read</code> permits an action to list the Dependabot alerts for the repository, and <code>security-events: write</code> allows an action to update the status of a code scanning alert. For more information, see " Repository permissions for 'Code scanning alerts' " and " Repository permissions for 'Dependabot alerts' " in "Permissions required for GitHub Apps."
statuses	Work with commit statuses. For example, <code>statuses: read</code> permits an action to list the commit statuses for a given reference. For more information, see " Permissions required for GitHub Apps. "

Defining access for the `GITHUB_TOKEN` scopes

You can define the access that the `GITHUB_TOKEN` will permit by specifying `read`, `write`, or `none` as the value of the available scopes within the `permissions` key.

```
permissions:  
  actions: read|write|none  
  checks: read|write|none  
  contents: read|write|none  
  deployments: read|write|none  
  id-token: read|write|none  
  issues: read|write|none
```

```
discussions: read|write|none
packages: read|write|none
pages: read|write|none
pull-requests: read|write|none
repository-projects: read|write|none
security-events: read|write|none
statuses: read|write|none
```

If you specify the access for any of these scopes, all of those that are not specified are set to `none`.

You can use the following syntax to define one of `read-all` or `write-all` access for all of the available scopes:

```
permissions: read-all
```

```
permissions: write-all
```

You can use the following syntax to disable permissions for all of the available scopes:

```
permissions: {}
```

Changing the permissions in a forked repository [↗](#)

You can use the `permissions` key to add and remove read permissions for forked repositories, but typically you can't grant write access. The exception to this behavior is where an admin user has selected the **Send write tokens to workflows from pull requests** option in the GitHub Actions settings. For more information, see "[Managing GitHub Actions settings for a repository](#)."

Setting the `GITHUB_TOKEN` permissions for all jobs in a workflow [↗](#)

You can specify `permissions` at the top level of a workflow, so that the setting applies to all jobs in the workflow.

Example: Setting the `GITHUB_TOKEN` permissions for an entire workflow [↗](#)

This example shows permissions being set for the `GITHUB_TOKEN` that will apply to all jobs in the workflow. All permissions are granted read access.

```
name: "My workflow"

on: [ push ]

permissions: read-all

jobs:
  ...
```

Setting the `GITHUB_TOKEN` permissions for a specific job [↗](#)

For a specific job, you can use `jobs.<job_id>.permissions` to modify the default

permissions granted to the `GITHUB_TOKEN` , adding or removing access as required, so that you only allow the minimum required access. For more information, see "[Automatic token authentication](#)."

By specifying the permission within a job definition, you can configure a different set of permissions for the `GITHUB_TOKEN` for each job, if required. Alternatively, you can specify the permissions for all jobs in the workflow. For information on defining permissions at the workflow level, see [permissions](#) .

Example: Setting the `GITHUB_TOKEN` permissions for one job in a workflow [↗](#)

This example shows permissions being set for the `GITHUB_TOKEN` that will only apply to the job named `stale` . Write access is granted for the `issues` and `pull-requests` scopes. All other scopes will have no access.

```
jobs:
  stale:
    runs-on: ubuntu-latest

    permissions:
      issues: write
      pull-requests: write

    steps:
      - uses: actions/stale@v5
```

Legal

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