About GitHub-hosted runners

In this article

GitHub offers hosted virtual machines to run workflows. The virtual machine contains an environment of tools, packages, and settings available for GitHub Actions to use.

Overview of GitHub-hosted runners

Runners are the machines that execute jobs in a GitHub Actions workflow. For example, a runner can clone your repository locally, install testing software, and then run commands that evaluate your code.

GitHub provides runners that you can use to run your jobs, or you can host your own runners. Each GitHub-hosted runner is a new virtual machine (VM) hosted by GitHub with the runner application and other tools preinstalled, and is available with Ubuntu Linux, Windows, or macOS operating systems. When you use a GitHub-hosted runner, machine maintenance and upgrades are taken care of for you.

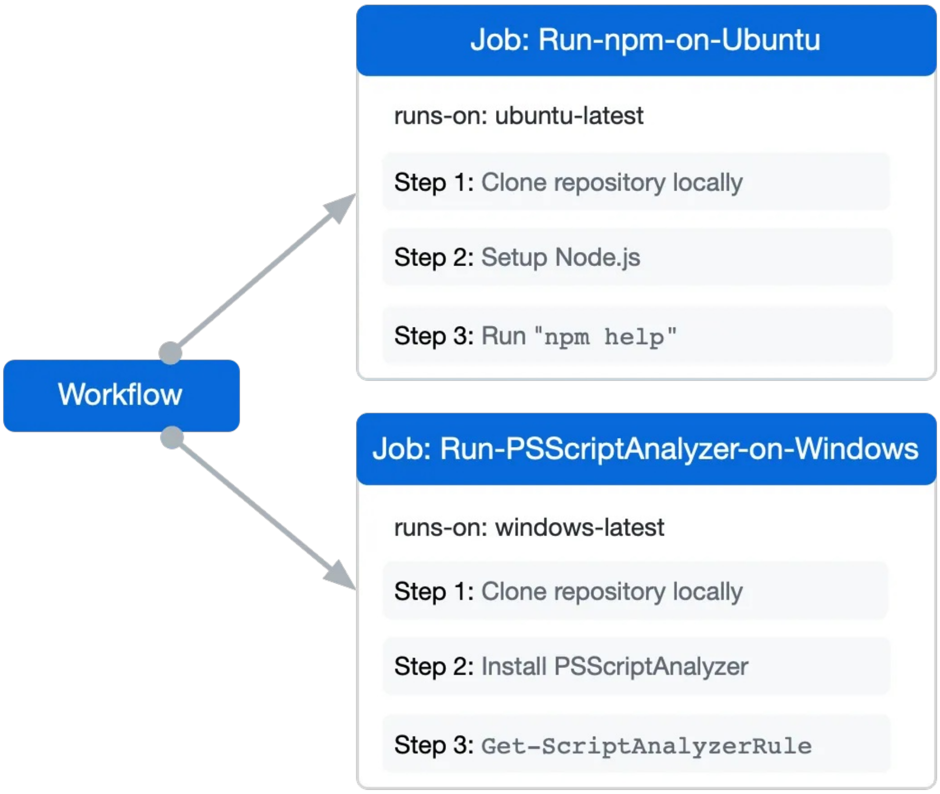
> **Note:** Entitlement minutes for GitHub-hosted runners are not available for managed user accounts. Enterprise Managed Users who would like to contribute to repositories in organizations they are not a member of can fork the organization repo, then open a pull request targeting the organization repository. This runs the workflows on the organization's GitHub-hosted runners. For more information, see "About Enterprise Managed Users."

Using a GitHub-hosted runner

To use a GitHub-hosted runner, create a job and use `runs-on` to specify the type of runner that will process the job, such as `ubuntu-latest`, `windows-latest`, or `macos-latest`. For the full list of runner types, see "Using GitHub-hosted runners." If you have `repo: write` access to a repository, you can view a list of the runners available to use in workflows in the repository. For more information, see "Viewing available runners for a repository."

When the job begins, GitHub automatically provisions a new VM for that job. All steps in the job execute on the VM, allowing the steps in that job to share information using the runner's filesystem. You can run workflows directly on the VM or in a Docker container. When the job has finished, the VM is automatically decommissioned.

The following diagram demonstrates how two jobs in a workflow are executed on two different GitHub-hosted runners.


Workflow
Job: Run-npm-on-Ubuntu
runs-on: ubuntu-latest
Step 1: Clone repository locally
Step 2: Setup Node.js
Step 3: Run "npm help"
Job: Run-PSScriptAnalyzer-on-Windows
runs-on: windows-latest
Step 1: Clone repository locally
Step 2: Install PSScriptAnalyzer
Step 3: Get-ScriptAnalyzerRule

The following example workflow has two jobs, named `Run-npm-on-Ubuntu` and `Run-PSScriptAnalyzer-on-Windows`. When this workflow is triggered, GitHub provisions a new virtual machine for each job.

- The job named `Run-npm-on-Ubuntu` is executed on a Linux VM, because the job's `runs-on:` specifies `ubuntu-latest`.
- The job named `Run-PSScriptAnalyzer-on-Windows` is executed on a Windows VM, because the job's `runs-on:` specifies `windows-latest`.

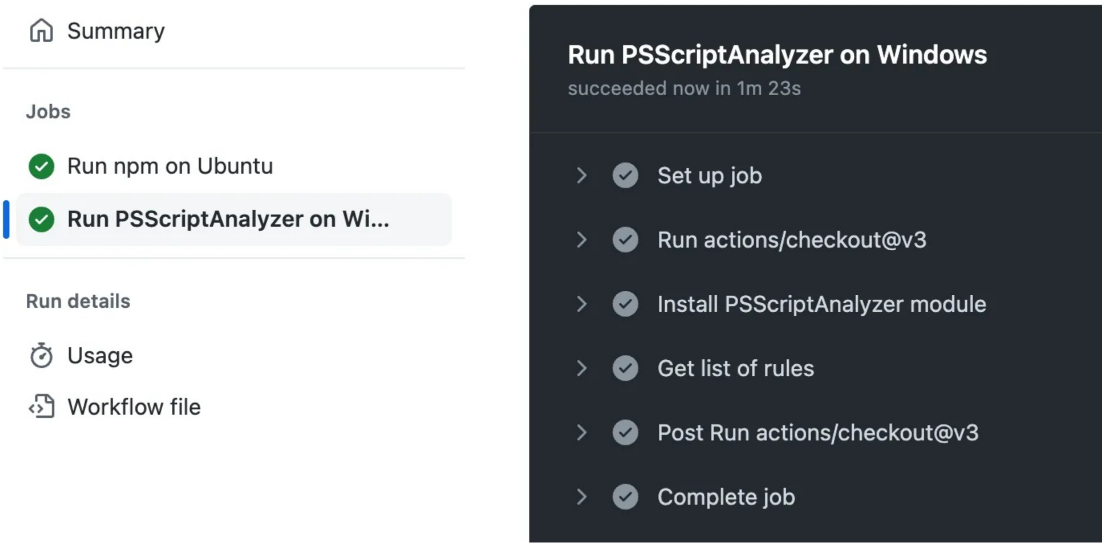
YAML

```yaml
name: Run commands on different operating systems
on:
  push:
    branches: [ main ]
  pull_request:
    branches: [ main ]

jobs:
  Run-npm-on-Ubuntu:
    name: Run npm on Ubuntu
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v4
      - uses: actions/setup-node@v3
        with:
          node-version: '14'
      - run: npm help

  Run-PSScriptAnalyzer-on-Windows:
    name: Run PSScriptAnalyzer on Windows
    runs-on: windows-latest
    steps:
      - uses: actions/checkout@v4
      - name: Install PSScriptAnalyzer module
        shell: pwsh
        run: |
          Set-PSRepository PSGallery -InstallationPolicy Trusted
          Install-Module PSScriptAnalyzer -ErrorAction Stop
      - name: Get list of rules
        shell: pwsh
        run: |
          Get-ScriptAnalyzerRule
```

While the job runs, the logs and output can be viewed in the GitHub UI:



Run PSScriptAnalyzer on Windows
succeeded now in 1m 23s

Set up job
Run actions/checkout@v3
Install PSScriptAnalyzer module
Get list of rules
Post Run actions/checkout@v3
Complete job

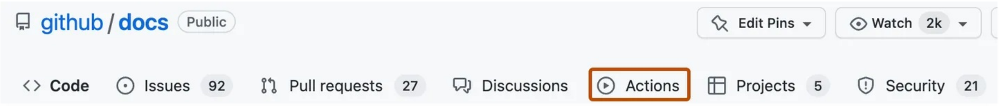
The GitHub Actions runner application is open source. You can contribute and file issues in the runner repository.

Viewing available runners for a repository

Note: This feature is currently in beta and subject to change.

If you have `repo: write` access to a repository, you can view a list of the runners available to the repository.

1. On GitHub.com, navigate to the main page of the repository.
2. Under your repository name, click **Actions**.



3. In the left sidebar, under the "Management" section, click **Runners**.
4. Review the list of available GitHub-hosted runners for the repository.
5. Optionally, to copy a runner's label to use it in a workflow, click ⋯ to the right of the runner, then click **Copy label**.

Note: Enterprise and organization owners with privileges to create runners have the option to create new runners from this page. If you are an enterprise or organization owner, click **New runner** at the top right of the list of runners to add runners to the repository. For more information, see "Managing larger runners" and "Adding self-hosted runners."

Supported runners and hardware resources

> **Note**: GitHub also offers larger runners, which are available in larger configurations for Linux, Windows, and macOS virtual machines. Autoscaling is enabled by default and optional dedicated IP addresses are available for Linux and Windows. For more information, see "About larger runners."

Virtual Machine	Processor (CPU)	Memory (RAM)	Storage (SSD)	OS (YAML workflow label)	Notes
Linux	2	7 GB	14 GB	`ubuntu-latest`, `ubuntu-22.04`, `ubuntu-20.04`	The `ubuntu-latest` label currently uses the Ubuntu 22.04 runner image.
Windows	2	7 GB	14 GB	`windows-latest`, `windows-2022`, `windows-2019`	The `windows-latest` label currently uses the Windows 2022 runner image.
macOS	3	14 GB	14 GB	`macos-latest`, `macos-12`, `macos-11`	The `macos-latest` workflow label currently uses the macOS 12 runner image.
macOS	4	14 GB	14 GB	`macos-13` [Beta]	N/A

> **Note:** The `-latest` runner images are the latest stable images that GitHub provides, and might not be the most recent version of the operating system available from the operating system vendor.

> **Warning:** Beta and Deprecated Images are provided "as-is", "with all faults" and "as available" and are excluded from the service level agreement and warranty. Beta Images may not be covered by customer support.

Workflow logs list the runner used to run a job. For more information, see "Viewing workflow run history."

Larger runners

In addition to the standard GitHub-hosted runners, GitHub offers customers on GitHub Team and GitHub Enterprise Cloud plans a range of managed virtual machines with more RAM, CPU, and disk space. These runners are hosted by GitHub and have the runner application and other tools preinstalled.

For more information, see "About larger runners."

Supported software

The software tools included in GitHub-hosted runners are updated weekly. The update process takes several days, and the list of preinstalled software on the `main` branch is updated after the whole deployment ends.

Preinstalled software

Workflow logs include a link to the preinstalled tools on the exact runner. To find this information in the workflow log, expand the `Set up job` section. Under that section, expand the `Runner Image` section. The link following `Included Software` will describe the preinstalled tools on the runner that ran the workflow.

For more information, see "Viewing workflow run history."

For the overall list of included tools for each runner operating system, see the links below:

- Ubuntu 22.04 LTS
- Ubuntu 20.04 LTS
- Windows Server 2022
- Windows Server 2019
- macOS 13
- macOS 12
- macOS 11

GitHub-hosted runners include the operating system's default built-in tools, in addition to the packages listed in the above references. For example, Ubuntu and macOS runners include `grep`, `find`, and `which`, among other default tools.

You can also view a software bill of materials (SBOM) for each build of the Windows and Ubuntu runner images. For more information, see "Security hardening for GitHub Actions."

Using preinstalled software

We recommend using actions to interact with the software installed on runners. This approach has several benefits:

- Usually, actions provide more flexible functionality like versions selection, ability to pass arguments, and parameters
- It ensures the tool versions used in your workflow will remain the same regardless of software updates

If there is a tool that you'd like to request, please open an issue at actions/runner-images. This repository also contains announcements about all major software updates on runners.

Installing additional software

You can install additional software on GitHub-hosted runners. For more information, see "Customizing GitHub-hosted runners".

Cloud hosts used by GitHub-hosted runners

GitHub hosts Linux and Windows runners on `Standard_DS2_v2` virtual machines in Microsoft Azure with the GitHub Actions runner application installed. The GitHub-hosted runner application is a fork of the Azure Pipelines Agent. Inbound ICMP packets are blocked for all Azure virtual machines, so ping or traceroute commands might not work. For more information about the `Standard_DS2_v2` resources, see "Dv2 and DSv2-series" in the Microsoft Azure documentation. GitHub hosts macOS runners in Azure data centers.

Workflow continuity

If GitHub Actions services are temporarily unavailable, then a workflow run is discarded if it has not been queued within 30 minutes of being triggered. For example, if a workflow is triggered and the GitHub Actions services are unavailable for 31 minutes or longer, then the workflow run will not be processed.

In addition, if the workflow run has been successfully queued, but has not been processed by a GitHub-hosted runner within 45 minutes, then the queued workflow run is discarded.

Administrative privileges

The Linux and macOS virtual machines both run using passwordless `sudo`. When you need to execute commands or install tools that require more privileges than the current user, you can use `sudo` without needing to provide a password. For more information, see the "Sudo Manual."

Windows virtual machines are configured to run as administrators with User Account Control (UAC) disabled. For more information, see "How User Account Control works" in the Windows documentation.

IP addresses

To get a list of IP address ranges that GitHub Actions uses for GitHub-hosted runners, you can use the GitHub REST API. For more information, see the `actions` key in the response of the "Meta" endpoint.

Windows and Ubuntu runners are hosted in Azure and subsequently have the same IP address ranges as the Azure datacenters. macOS runners are hosted in GitHub's own macOS cloud.

Since there are so many IP address ranges for GitHub-hosted runners, we do not recommend that you use these as allowlists for your internal resources. Instead, we recommend you use larger runners with a static IP address range, or self-hosted runners. For more information, see "About larger runners" or "About self-hosted runners."

The list of GitHub Actions IP addresses returned by the API is updated once a week.

The `etc/hosts` file

GitHub-hosted runners are provisioned with an `etc/hosts` file that blocks network access to various cryptocurrency mining pools and malicious sites. Hosts such as MiningMadness.com and cpu-pool.com are rerouted to localhost so that they do not present a significant security risk.

File systems

GitHub executes actions and shell commands in specific directories on the virtual machine. The file paths on virtual machines are not static. Use the environment variables GitHub provides to construct file paths for the `home`, `workspace`, and `workflow` directories.

Directory	Environment variable	Description
`home`	`HOME`	Contains user-related data. For example, this directory could contain credentials from a login attempt.
`workspace`	`GITHUB_WORKSPACE`	Actions and shell commands

		execute in this directory. An action can modify the contents of this directory, which subsequent actions can access.
`workflow/event.json`	`GITHUB_EVENT_PATH`	The `POST` payload of the webhook event that triggered the workflow. GitHub rewrites this each time an action executes to isolate file content between actions.

For a list of the environment variables GitHub creates for each workflow, see "Variables."

Docker container filesystem

Actions that run in Docker containers have static directories under the `/github` path. However, we strongly recommend using the default environment variables to construct file paths in Docker containers.

GitHub reserves the `/github` path prefix and creates three directories for actions.

- `/github/home`
- `/github/workspace` - **Note:** GitHub Actions must be run by the default Docker user (root). Ensure your Dockerfile does not set the `USER` instruction, otherwise you will not be able to access `GITHUB_WORKSPACE`.
- `/github/workflow`

Further reading

- "Managing billing for GitHub Actions"
- You can use a matrix strategy to run your jobs on multiple images. For more information, see "Using a matrix for your jobs."