



# GitHub Actions

## Fundamentals

Presented by GitHub Professional Services

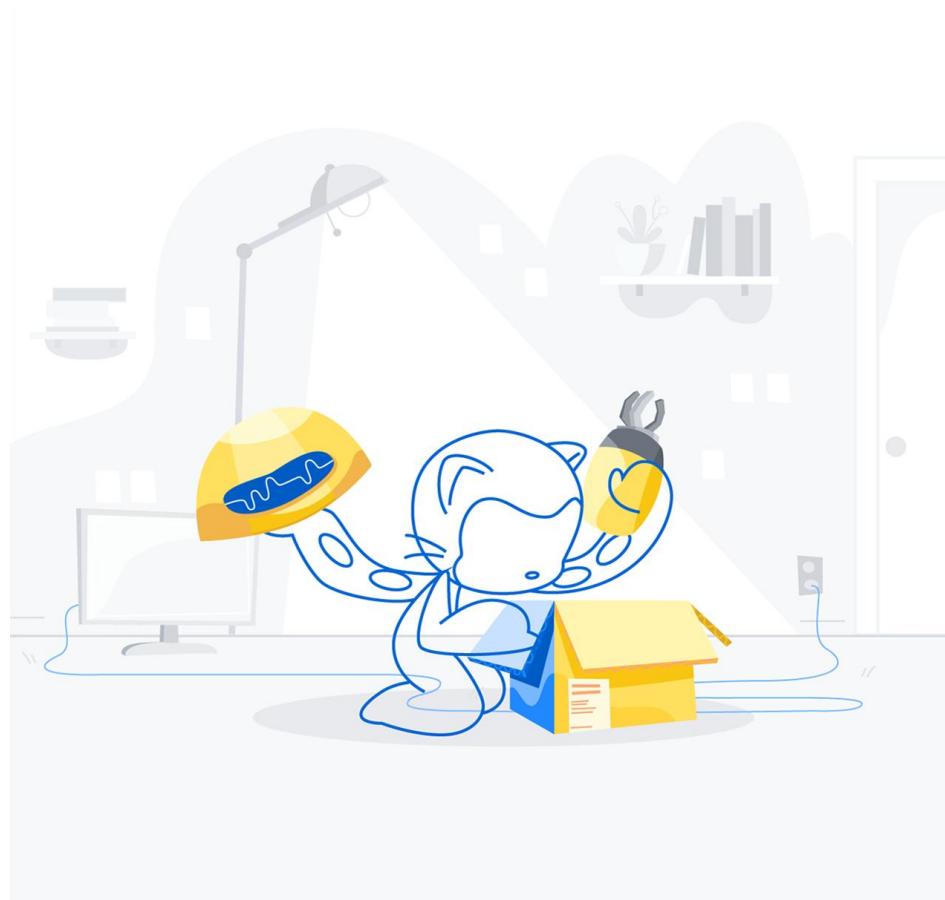
# Objectives



- Understand the **basic components** of GitHub Actions and its **use cases**
- Understand the GitHub Actions **syntax**, both for actions and workflows
- Know how to navigate GitHub Actions **GUI and documentation**
- Understand how to leverage actions written by the **community**
- Create **custom actions**
- Automate both **CI/CD and non-CI/CD** use cases
- Know how to use **environments and secrets**
- Understand how to **migrate** to GitHub Actions from a different CI/CD system
- Understand the differences between **GitHub-hosted and self-hosted runners**
- Understand **best practices** related to GitHub Actions

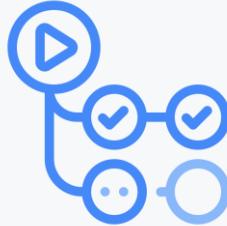
# Agenda

- Introduction to GitHub Actions
- Workflow syntax
- Environments and secrets
- Managing workflows & Actions
- Building Actions
- Migration
- Runners
- CI/CD workflows 
- Demos!



# Introduction

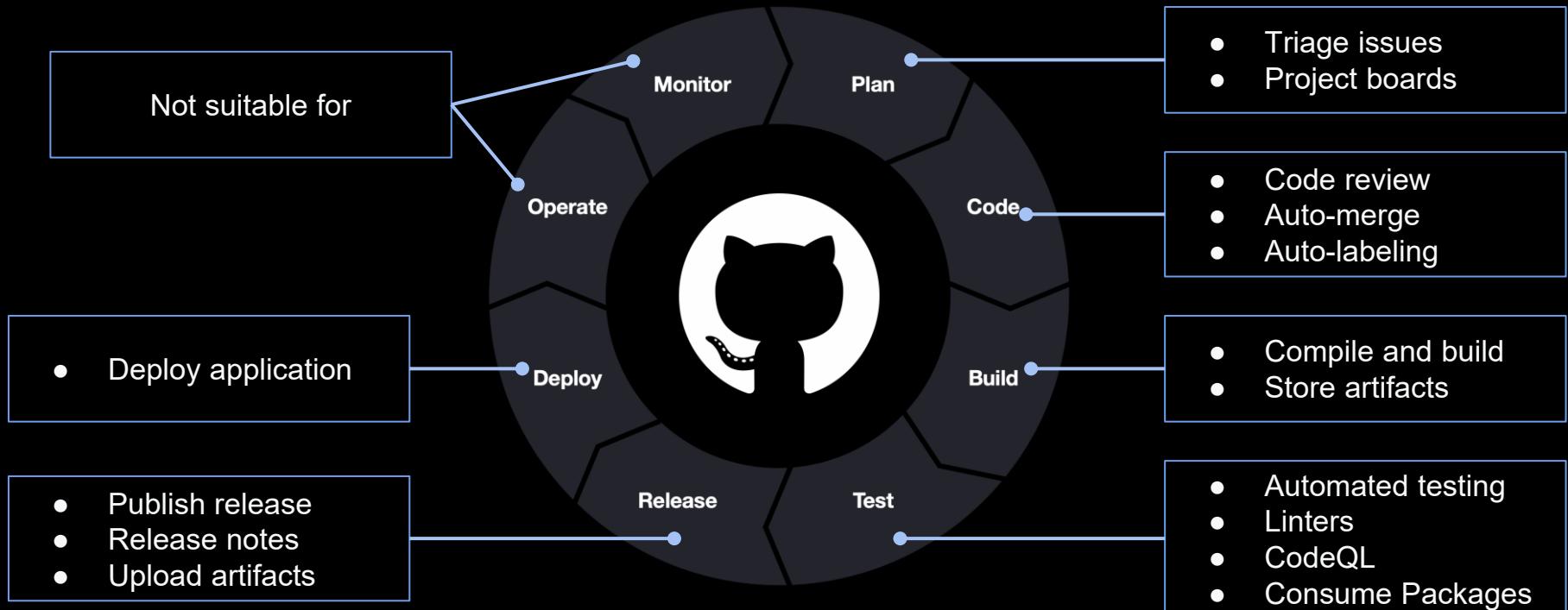
# What is GitHub Actions



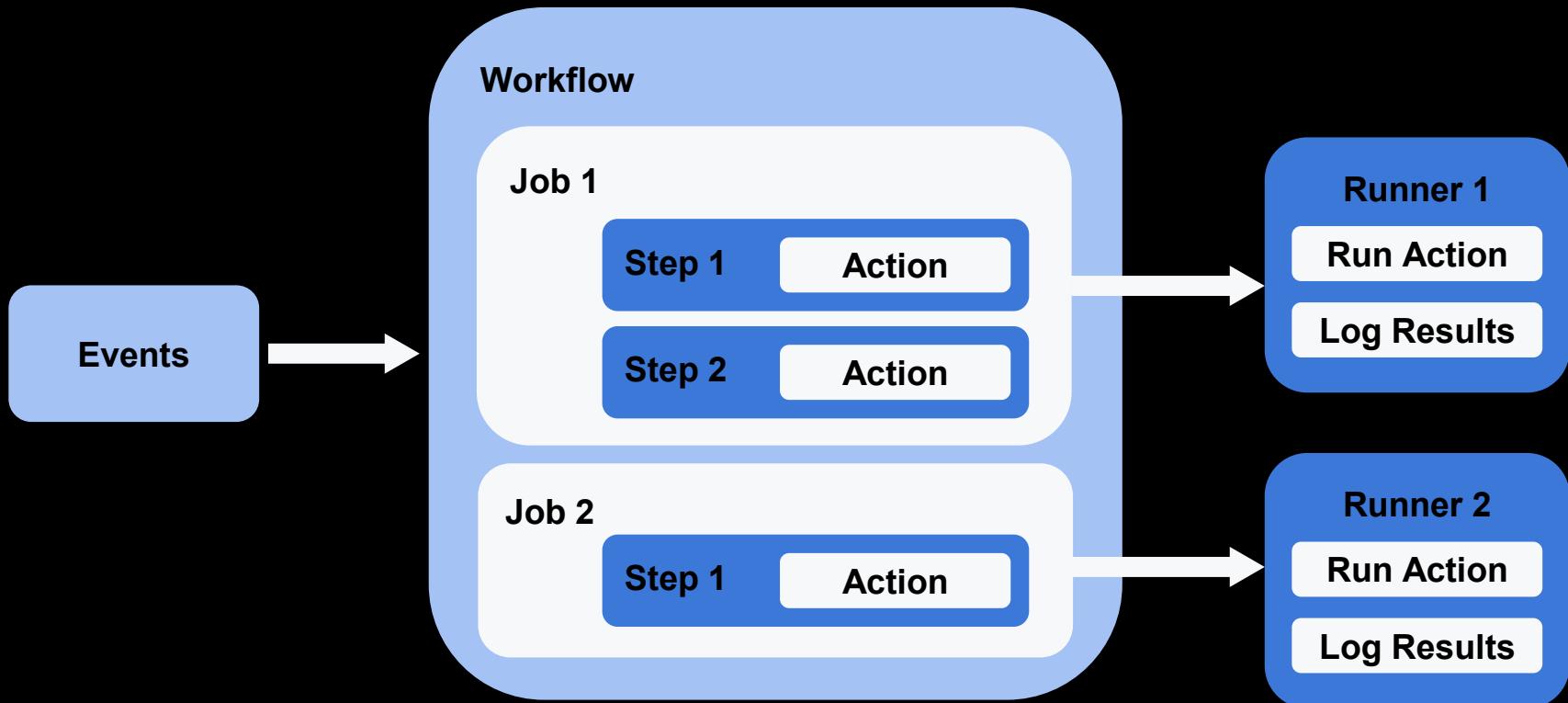
GitHub Actions is a GitHub product that allows you to **automate your workflows**.

- Workflows stored as yml files
- Fully integrated with GitHub
- Respond to GitHub events
- Live logs and visualized workflow execution
- Community-powered workflows
- GitHub-hosted or self-hosted runners
- Built-in secret store

# Use cases across your SDLC



# Key components



# Workflow syntax

# Basic syntax

```
./.github/workflows/workflow-file-name.yml
```

```
name: Super Linter workflow
```

events → `on:`  
          `push:`

jobs → `jobs:`  
          `lint:`  
            `name: Lint Code Base`

runner → `runs-on: ubuntu-latest`

steps → `steps:`

actions →     `- uses: actions/checkout@v2`

`- uses: github/super-linter@v3`  
              `env:`

secrets →     `GITHUB_TOKEN: ${{ secrets.GITHUB_TOKEN }}`

# Events

## Webhook events

- Pull request
- Issues
- Push
- Release
- ...

events

```
name: Super Linter workflow
```

```
on:  
  issues:  
    types: [closed, reopened]
```

## Scheduled events

## Manual events

```
jobs:  
  lint:  
    name: Lint Code Base  
  
    runs-on: ubuntu-latest
```

```
steps:
```

```
- uses: actions/checkout@v2
```

```
- uses: github/super-linter@v3
```

```
env:
```

```
GITHUB_TOKEN: ${{ secrets.GITHUB_TOKEN }}
```

# Events

## Webhook events

- Pull request
- Issues
- Push
- Release
- ...

## Scheduled events

## Manual events

events

```
name: Super Linter workflow
```

```
on:
```

```
schedule:
```

```
- cron: '30 6 * * 5' # every Friday 06:30 UTC
```

```
jobs:
```

```
lint:
```

```
    name: Lint Code Base
```

```
runs-on: ubuntu-latest
```

```
steps:
```

```
- uses: actions/checkout@v2
```

```
- uses: github/super-linter@v3
```

```
env:
```

```
GITHUB_TOKEN: ${{ secrets.GITHUB_TOKEN }}
```

# Events

## Webhook events

- Pull request
- Issues
- Push
- Release
- ...

## Scheduled events

## Manual events

- workflow\_dispatch
- repository\_dispatch

events

```
name: Super Linter workflow
```

```
on:  
  workflow_dispatch:
```

```
jobs:  
  lint:  
    name: Lint Code Base
```

```
runs-on: ubuntu-latest
```

```
steps:
```

```
- uses: actions/checkout@v2
```

```
- uses: github/super-linter@v3
```

```
env:
```

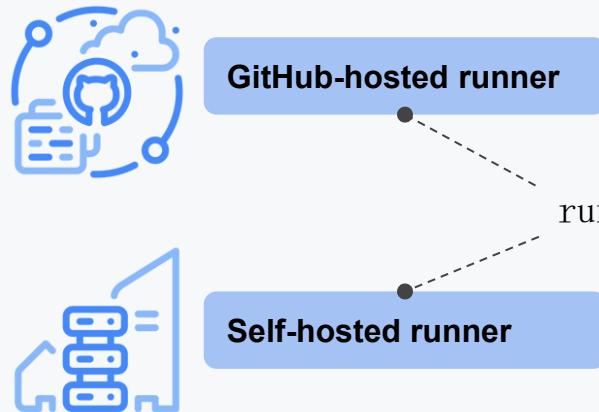
```
GITHUB_TOKEN: ${{ secrets.GITHUB_TOKEN }}
```

Event ▾ Status ▾ Branch ▾ Actor ▾

This workflow has a workflow\_dispatch event trigger.

Run workflow ▾

# Runners



```
name: Super Linter workflow

on:
  push:

jobs:
  lint:
    name: Lint Code Base
    runs-on: ubuntu-latest

    steps:
      - uses: actions/checkout@v2
      - uses: github/super-linter@v3
        env:
          GITHUB_TOKEN: ${{ secrets.GITHUB_TOKEN }}
```

# Runners



## GitHub-hosted runner

- OS: ubuntu, windows, or macOS
- Ephemeral
- 2-core CPU (macOS: 3-core)
- 7 GB RAM (macOS: 14 GB)
- 14 GB SSD disk space
- Software installed: wget, GH CLI, AWS CLI, Java, ...
- Not currently available on GHES



# Runners



## Self-hosted runner

- Custom hardware config
- Run on OS not supported on GitHub-hosted runner
- Reference runner using custom labels
- Can be grouped together
- Control which organizations/repositories have access to which runners/runner groups
- Do not use with public repositories!

runner →

```
name: Super Linter workflow

on:
  push:

jobs:
  lint:
    name: Lint Code Base

    runs-on: [self-hosted, linux, ARM64]

    steps:
      - uses: actions/checkout@v2
      - uses: github/super-linter@v3
        env:
          GITHUB_TOKEN: ${{ secrets.GITHUB_TOKEN }}
```

# Actions

Reusable units of code that can be referenced in a workflow

GitHub runs them in Node.js runtime, or in Docker containers

Reference an Action, or run scripts directly

Can be referenced in three ways:

- Public repository
- The same repository as your workflow (local actions)
- A published Docker container image on DockerHub

```
name: Super workflow
```

```
on:  
  push:
```

```
jobs:  
  lint:  
    name: Lint Code Base
```

```
runs-on: ubuntu-latest
```

```
steps:
```

```
  - run: echo "Hello World"
```

```
  - uses: actions/checkout@v2  
  
  - uses: github/super-linter@v3  
    env:  
      GITHUB_TOKEN: ${{ secrets.GITHUB_TOKEN }}
```

```
  - uses: ./path/to/action
```

```
  - uses: docker://alpine:3.8
```

script →

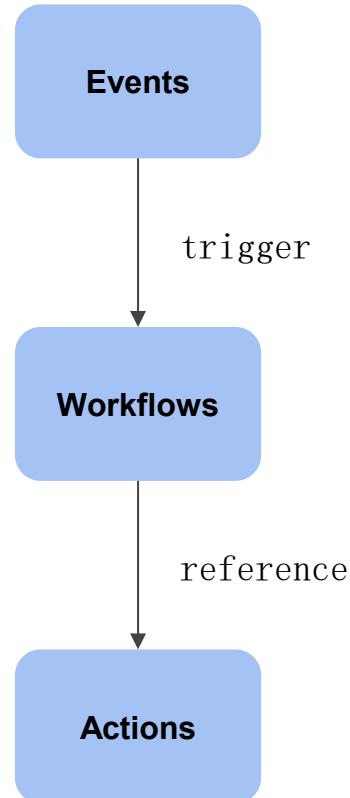
public actions →

local action →

docker image →

# Quick summary

- Events trigger workflows, e.g. a push to a branch
- Workflows contain one or more jobs, which contains one or more steps
- These steps can reference actions or execute commands
- The term “*Github Actions*” include all components, not just the Actions themselves



# GitHub Marketplace

- Discover open-source Actions across multiple domains
- ~9,000 Actions (and counting...) 
- Verified creators
- Reference these Actions directly in your workflow
- Integrated into the GitHub editor

## Extend GitHub

Add tools to help you build and grow

[Explore apps](#)



### Types

- Apps
  - Actions
- 
- Categories
  - API management
  - Chat
  - Code quality
  - Code review
  - Continuous integration
  - Dependency management
  - Deployment
  - IDEs
  - Learning
  - Localization
  - Mobile
  - Monitoring
  - Project management
  - Publishing
  - Recently added
  - Security
  - Support

Search for apps and actions

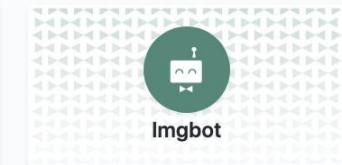
### Recommended for you



Codecov | Code Coverage

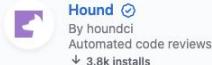


Stale



Imgbot

### Trending



Hound

By houndci

Automated code reviews

↓ 3.8k installs



Moesif API Insights

By Moesif

Understand API usage and take action with user-centric API observability

↓ 698 installs



Octobox

By octobox

Spend less time managing your GitHub notifications

↓ 4.4k installs



WhiteSource Bolt

By whitesource

Detect open source vulnerabilities in real time with suggested fixes for quick remediation

↓ 3.6k installs

# Starter workflows

- Preconfigured for specific languages and frameworks
- GitHub analyzes your code and suggests the workflows based on your language and framework
- For GHES 3.x: A number of starter workflows come pre-packaged with the release.

**Deploy Node.js to Azure Web App**  
By Microsoft Azure

Build a Node.js project and deploy it to an Azure Web App.

[Set up this workflow](#)

 actions/starter-workflows Deployment ●

**Deploy to Amazon ECS**  
By Amazon Web Services

Deploy a container to an Amazon ECS service powered by AWS Fargate or Amazon EC2.

[Set up this workflow](#)

 actions/starter-workflows Deployment ●

**Ruby**  
By GitHub Actions

Build and test a Ruby project with Rake.

[Set up this workflow](#)

 actions/starter-workflows Ruby ●

**Python application**  
By GitHub Actions

Create and test a Python application.

[Set up this workflow](#)

 actions/starter-workflows Python ●

**Clojure**  
By GitHub Actions

Build and test a Clojure project with Leiningen.

[Set up this workflow](#)

 actions/starter-workflows Clojure ●

**Node.js**  
By GitHub Actions

Build and test a Node.js project with npm.

[Set up this workflow](#)

 actions/starter-workflows JavaScript ●

**Publish Docker Container**  
By GitHub Actions

Build, test and push Docker image to GitHub Packages.

[Set up this workflow](#)

 actions/starter-workflows Dockerfile ●

(...)

# Workflow logs

stebje-actions-packages / demo-publish Private

Watch 0 Star 0 Fork 0

Code Issues Pull requests 1 Actions Projects Wiki Security Insights Settings

Update README.md Chatops-cmd #8 Re-run jobs ⋮

Summary

Jobs

publish-command

**publish-command**  
succeeded 25 minutes ago in 5s

Search logs ⚙

**Set up job**

```
1 Current runner version: '2.278.0'
2 ▶ Operating System
3 ▶ Virtual Environment
11 ▶ GITHUB_TOKEN Permissions
24 Prepare workflow directory
25 Prepare all required actions
26 Getting action download info
27 Download action repository 'peter-evans/slash-command-dispatch@v2'
```

**Slash Command Dispatch**

```
1 ▼ Run peter-evans/slash-command-dispatch@v2
2   with:
3     token: ***
4     commands: publish
5
6     reactions: true
7     issue-type: pull-request
8     permission: maintain
9     reaction-token: ***
10    allow-edits: false
11    repository: stebje-actions-packages/demo-publish
12    event-type-suffix: -command
13    dispatch-type: repository
14 Using configuration from yaml inputs.
15 Command 'publish' to be dispatched.
16 Command 'publish' dispatched to 'stebje-actions-packages/demo-publish' with event type 'publish-command'.
```

# Advanced syntax

Syntax element	Description
permissions	Set workflow permissions for GITHUB_TOKEN
env	Set environment variables for all run steps
defaults	Set the shell and working directory for the run
concurrency	Manage workflows running concurrently
needs	Make jobs dependent of each other. Share outputs
if	Check whether a job should run based on variables. success() always() cancelled() failure()
timeout	Limit runtime
continue-on-error	Handle termination of workflows
services	Create sidecar docker images for integration dependencies
container	Use a container for the steps execution

# Function expressions

Syntax element	Description
contains	Check if a string is contained in another
startsWith/endsWith	Check start/end of a string
format	Format outputs
join	Join arrays into strings
toJSON/fromJSON	Make string JSON and JSON strings
hashFiles	Create a hash from an input file. Useful for caching
always/success/failure/cancelled	Workflow statuses. Useful for conditional runs



Demo

# Environments and secrets

# Environments

- Control deployments
- Add gated deployments with approvals
- Control secrets
- Review all deployments to an env
- Navigate directly to urls for deployments
- Fully integrated with the checks API (previously called deployment API)
- Supports matrix for gated deployments

## Environments / Configure Development

### Environment protection rules

Can be used to configure manual approvals and timeouts.

#### Required reviewers

Specify people or teams that may approve workflow runs when they access this environment.

Add up to 6 more reviewers

Search for people or teams...

#### Wait timer

Set an amount of time to wait before allowing deployments to proceed.

15 minutes

[Save protection rules](#)

### Deployment branches

Can be used to limit what branches can deploy to this environment using branch name patterns.

All branches ▾

### Environment secrets

Secrets are encrypted environment variables. They are accessible only by GitHub Actions in the context of this environment.

[⊕ Add Secret](#)

# GitHub Secret store

- Built-in secret store
- Encrypted
  - LibSodium sealed box
- Use directly from your workflow
- Redacted in workflow logs
- API support
- Organization / repository / environment secrets

The screenshot shows the GitHub repository settings for 'stebje-actions-packages / demo-ci' (Private). The left sidebar has 'Secrets' selected. The main area displays four types of secrets:

- Actions secrets**: Secrets are environment variables that are encrypted. Anyone with collaborator access to this repository can use these secrets for Actions. Secrets are not passed to workflows that are triggered by a pull request from a fork. [Learn more](#).
- Environment secrets**: MY\_ENV\_SECRET (TEST\_ENV) - Updated 1 minute ago. Buttons: Manage environment.
- Repository secrets**: MY\_REPO\_SECRET - Updated 2 minutes ago. Buttons: Update, Remove.
- Organization secrets**: MY\_ORG\_SECRET - Updated 1 minute ago. Buttons: Manage organization secrets.

Text at the bottom: Secrets can also be created at the organization level and authorized for use in this repository.

# Types of secrets

## ● Environment secrets

- Scoped to a single environment
- The secret is not accessible by workflow unless the deployment to that environment is approved

## ● Repository secrets

- Scoped to a single repository
- Can override org-level secrets

## ● Organization secrets

- Managed at org-level
- Can be scoped to specific repositories

The screenshot shows a GitHub repository named "stebje-actions-packages / demo-ci" with a "Private" status. The top navigation bar includes "Watch 0", "Star 0", "Fork 0", and other repository management links.

The left sidebar lists repository settings categories: Options, Manage access, Security & analysis, Branches, Webhooks, Notifications, Integrations, Deploy keys, Autolink references, Actions, Environments, Secrets (which is currently selected), Actions, Codespaces, Dependabot, and Pages.

The main content area is titled "Actions secrets" and contains the following text: "Secrets are environment variables that are **encrypted**. Anyone with **collaborator** access to this repository can use these secrets for Actions. Secrets are not passed to workflows that are triggered by a pull request from a fork. [Learn more](#)".

Three sections are highlighted with red boxes:

- Environment secrets**: Contains secrets "MY\_ENV\_SECRET" and "TEST\_ENV". Last updated 1 minute ago. Includes "Manage environment" button.
- Repository secrets**: Contains secret "MY\_REPO\_SECRET". Last updated 2 minutes ago. Includes "Update" and "Remove" buttons.
- Organization secrets**: Contains secret "MY\_ORG\_SECRET". Last updated 1 minute ago. Includes "Manage organization secrets" button.

A note at the bottom states: "Secrets can also be created at the organization level and authorized for use in this repository."

# Using secrets in workflows

- All secrets can be accessed using the same syntax; `${{ secrets.<SECRET_NAME> }}`
- Every workflow run provisions a **GITHUB\_TOKEN** secret by default
  - Scoped to a single repository
  - Granular permissions
  - Can't trigger other workflows
- Marketplace Actions exist for integration with other secret stores



## Vault Secrets

By hashicorp

A Github Action that allows you to consume HashiCorp Vault™ secrets as secure environment variables



## Azure key vault - Get Secrets

By Azure

Get Secrets from Azure Key Vault instance and set as output variables.  
[github.com/azure/actions](https://github.com/azure/actions)

```
name: Pull request labeler

on:
  pull_request:

jobs:
  triage:
    runs-on: ubuntu-latest

steps:
  - uses: actions/labeleder@v2
    with:
      repo-token: ${{ secrets.GITHUB_TOKEN }}

  - uses: myAction@v1
    with:
      mySecret: ${{ secrets.MY_SECRET }}
```



Demo

# Managing workflows & Actions

# Actions policies

- Configure Actions policies on enterprise / organization / repository level
  - Which Actions are allowed
  - Artifact retention period
  - Running workflows from fork PRs
  - Permissions of GITHUB\_TOKEN

The screenshot shows the GitHub Actions settings interface. On the left is a sidebar with links: Account settings, Profile, Billing & plans, Member privileges, Organization security, Security & analysis, Verified & approved domains, Audit log, Webhooks, Third-party access, Installed GitHub Apps, Scheduled reminders, Repository topics, Repository defaults, Deleted repositories, Projects, Teams, **Actions**, General, Runners, Packages, Secrets, Developer settings, and Moderation settings. The 'Actions' link is highlighted with a red border.

**Actions permissions**

Policies

Choose which repositories are permitted to use GitHub Actions.

**Allow all actions**  
Any action can be used, regardless of who authored it or where it is defined.

**Allow local actions only**  
Only actions defined in a repository within the enterprise can be used.

**Allow select actions**  
Only actions that match specified criteria, plus actions defined in a repository within the enterprise, can be used. [Learn more about allowing specific actions to run.](#)

**Save**

**Artifact and log retention**

This is the default duration that repositories will retain all artifacts and logs. Your enterprise administrator has set a maximum limit of **90 days**.

90 days **Save**

**Fork pull request workflows**

These settings apply to private repositories. Repository administrators will only be able to change the settings that are *enabled* here.

**Run workflows from fork pull requests**  
This tells Actions to run workflows from pull requests originating from repository forks. Note that doing so will give maintainers of those forks the ability to use tokens with read permissions on the source repository.

**Save**

**Workflow permissions**

Choose the default permissions granted to the GITHUB\_TOKEN when running workflows in this organization. You can specify more granular permissions in the workflow using YAML. [Learn more](#).

Repository administrators will only be able to change the default permissions to a more restrictive setting.

**Read and write permissions**  
Workflows have read and write permissions in the repository for all scopes.

**Read repository contents permission**  
Workflows have read permissions in the repository for the contents scope only.

**Save**

# Sharing workflows in an organization

- Create GitHub actions starter templates in .github repository to share workflows
- (Upcoming) Organization workflow execution. Open source concept:  
<https://github.com/SvanBoxel/organization-workflows>



# Reusable Workflows

- A workflow that can be called by other workflows
- Public repository or same repository
- `workflow_call` event
- Accepts inputs

```
on:  
  workflow_call:  
    inputs:  
      username:  
        required: true  
        type: string  
    secrets:  
      envPAT:  
        required: true
```



# Sharing private actions

Use GitHub packages and ghcr.io to share actions using docker execution and **package registry** permissions

Use a **GitHub App** to clone actions from:

- Actions in different repositories
- Actions monorepo
- Actions separate organization

```
jobs:  
  do-something:  
    runs-on: ubuntu-latest  
  
steps:  
  - name: Generate app installation token  
    id: app  
    uses: peter-murray/workflow-application-token-action@v1  
    with:  
      application_id: ${{ secrets.APP_ID }}  
      application_private_key: ${{ secrets.PRIV_KEY }}  
  
  - name: Checkout private repository  
    id: checkout_repo  
    uses: actions/checkout@v2  
    with:  
      repository: my-org/repo  
      path: path/to/privateAction  
      token: ${{ steps.app.outputs.token }}
```

# Sharing organization actions

- Enabled for Organization by default
- GHES
  - Internal
  - Private
- Configurable at Repository hosting Action

## Access

Control how this repository is used by GitHub Actions workflows in other repositories. [Learn more about allowing other repositories to access to Actions components in this repository.](#)

**Not accessible**

Workflows in other repositories cannot access this repository.

**Accessible from repositories in the 'US-SouthOU-Demo' organization**

Workflows in other repositories that are part of the 'US-SouthOU-Demo' organization can access the actions and reusable workflows in this repository. Access is allowed only from private or internal repositories.

**Accessible from repositories in the 'MSFT Demo Accounts (Internal)' enterprise**

Workflows in other repositories that are part of the 'MSFT Demo Accounts (Internal)' enterprise can access the actions and reusable workflows in this repository. Access is allowed only from private or internal repositories.

[Save](#)

# Caching

Optimizing your workflow performance with caching:

- Temporarily save files between workflow runs
- 5GB max cache size per repo
- 7 days retention
- Scoped to key and branch
- Avoid caching sensitive data
- Never cache sensitive data



## [Caching dependencies](#) to speed up workflows

Caching can help with speeding up workflows when you need to install dependencies. NPM, Python, Ruby, etc... these are simple examples of applications that require dependencies to be built. But there are more complex scenarios, such as Java, C/C++ and modularized microservices that often require downstream artifacts. Caching can speed up your builds when your dependencies have not changed

# Best practices on Actions in an organization

- Use the **GITHUB\_TOKEN** when possible, as a second option GitHub Apps
- Limit token permissions
- Run only **trusted actions**
- Protect your secrets with **environments**
- Create **starter workflows** for reusability



Demo

# Building Actions

# Writing your own Actions

- 3 types of Actions
  - JavaScript
  - Docker
  - Composite run step
- Metadata defined in `action.yml` file
  - Inputs
  - Outputs
  - Branding
  - Pre-/post-scripts
  - ...

```
./path/to/action/action.yml
```

```
name: "Hello Action"
description: "Greet someone"
author: "octocat@github.com"

inputs:
  MY_NAME:
    description: "Who to greet"
    required: true
    default: "World"

outputs:
  GREETING:
    description: "Full greeting"

runs:
  using: "docker"
  image: "Dockerfile"

branding:
  icon: "mic"
  color: "purple"
```

# Using the GitHub API

- REST API (v3)

- Libraries available for most languages

- Octokit

- GraphQL (v4)

- The future of the GitHub API
  - A query language allowing granular control of request and response

The screenshot shows a browser window displaying the GitHub API documentation for the `repos.createRelease` endpoint. The URL is `octokit.github.io/rest.js/v18#repos-create-release`. On the left, a sidebar lists various repository-related endpoints. The main content area is titled "Create a release" and provides a brief description: "Users with push access to the repository can create a release. This endpoint triggers notifications. Creating content too quickly using this endpoint may result in abuse rate limiting. See ['Abuse rate limits'](#) and ["Dealing with abuse rate limits"](#) for details." Below this is a "Parameters" table:

name	required	description
owner	yes	
repo	yes	
tag_name	yes	The name of the tag.
target_commitish	no	Specifies the commitish value that determines where the Git tag is created from. Can be any branch or commit SHA. Unused if the Git tag already exists. Default: the repository's default branch (usually <code>master</code> ).

On the far right, a dark sidebar displays the corresponding Octokit library code:

```
octokit.rest.repos.createRelease({  
  owner,  
  repo,  
  tag_name,  
});
```

# Writing your own Actions

## Best Practices

- Design for reusability
- Write tests
- Versioning
- Documentation
- Proper action.yml metadata
- [github.com/actions/toolkit](https://github.com/actions/toolkit)
- Publish your Action to the Marketplace 





actions / **github-script**

Watch

15

Star

1.1k

Fork

123

Code

Issues 7

Pull requests

Actions

Wiki

Security

Insights

#### Use this GitHub Action with your project

Add this Action to an existing workflow or create a new one.

[View on Marketplace](#)

main

Branches

Tags

Go to file

Add file

Code

joshmgross Merge pull request #137 from actions/joshgross/update-actions... · a3e7071 28 days ago · 247 commits

.github	<a href="#">Workflow syntax error</a>	last month
.licenses/npm	Update license for @actions/core	28 days ago
.vscode	Check in .vscode	12 months ago
_test_	Add ESLint and Prettier	12 months ago
dist	Update @actions/core to 1.2.7	28 days ago
docs	Update development.md	6 months ago
src	Remove require search fallback	last month
types	Pass nativeRequire, as well	last month
.eslintrc.yml	Add ESLint and Prettier	12 months ago
.gitattributes	Mark licenses as generated	10 months ago
.gitignore	Check in .vscode	12 months ago
.licensed.yml	Add production licenses with licensed	10 months ago
.prettierrc.yml	Add ESLint and Prettier	12 months ago

#### About

Write workflows scripting the GitHub API in JavaScript

[javascript](#) [github-api](#) [actions](#)

[Readme](#)

[MIT License](#)

#### Releases 32

[Update @actions/core pac...](#) Latest 28 days ago

+ 31 releases

#### Contributors 28



+ 17 contributors

#### Languages

TypeScript 100.0%



Demo

# Migration

# Runners

# Runners

## GitHub-hosted

- Receive automatic updates for the operating system, pre-installed packages and tools, and the self-hosted runner application.
- Are managed and maintained by GitHub.
- Provide a clean instance for every job execution.
- Use free minutes on your GitHub plan, with per-minute rates applied after surpassing the free minutes.

## Self-hosted

- Receive automatic updates for the self-hosted runner application only. You are responsible updating the operating system and all other software.
- Can use cloud services or local machines that you already pay for.
- Are customizable to your hardware, operating system, software, and security requirements.
- Don't need to have a clean instance for every job execution.
- Are free to use with GitHub Actions, but you are responsible for the cost of maintaining your runner machines.

# Larger Runners (GitHub Hosted)

- Ubuntu & Linux
- CPU Cores / RAM
- Auto-Scaling
- Runner Groups
- Static IP Addresses
- Public Preview

Runners / Create GitHub-hosted runner

Name

Runner image



Ubuntu



Windows Server

Ubuntu version

GitHub images are kept up to date and secure, containing all the tools you need to get started building and testing your applications. [Learn more about images](#).

"Latest" tag matches with standard GitHub-hosted runners latest tag for the images. [Learn more about latest tags](#).

Latest (20.04)

Runner size

4-cores - 16 GB RAM - 150 GB HDD

Auto-scaling

Maximum runners

50

Runners will not auto-scale above the maximum.  
Use this setting to limit your cost.

Runner groups 

The runner group will determine which organizations and repositories can use the runner. [Learn more about runner groups](#).

Default

All repositories, excluding public repositories

Labels

Labels are values used with the runs-on: key in your workflow's YAML to send jobs to specific runners. [Learn more about labels](#).



Networking

Assign a unique & static public IP address range for this runner

All instances of this GitHub-hosted runner will be assigned a static IP from a range unique to this runner. [Learn more about networking for runners](#).

You have used 0 out of 10 static public IP addresses available on your account.

Create runner

# Adding self-hosted runners

- Configure on enterprise / organization / repository level
- Download and extract the scripts
- Configure and authenticate the runner with the token
- Start listening for jobs
- For GHES: Blob storage must be provided (Azure Blob storage, Amazon S3, MinIO)

The screenshot shows the GitHub Actions & Packages interface. On the left, there's a sidebar with various settings like Account settings, Profile, Billing & plans, Member privileges, Organization security, Security & analysis, Verified & approved domains, Audit log, Webhooks, Third-party access, Installed GitHub Apps, Scheduled reminders, Repository topics, Repository defaults, Deleted repositories, Projects, Teams, Actions (which is highlighted), General, and Runners.

The main content area is titled "Actions / Add self-hosted runner". It contains instructions: "Adding a self-hosted runner requires that you download, configure, and execute the GitHub Actions Runner. By downloading and configuring the GitHub Actions Runner, you agree to the [GitHub Terms of Service](#) or [GitHub Corporate Terms of Service](#), as applicable." Below this are dropdowns for "Operating System: Linux" and "Architecture: X64".

The "Download" section provides a shell script for Linux:

```
# Create a folder
$ mkdir actions-runner && cd actions-runner
# Download the latest runner package
$ curl -o actions-runner-linux-x64-2.278.0.tar.gz -L
https://github.com/actions/runner/releases/download/v2.278.0/actions-runner-linux-x64-2.278.0.tar.gz
# Extract the installer
$ tar xzf ./actions-runner-linux-x64-2.278.0.tar.gz
```

The "Configure" section provides a shell script for configuration:

```
# Create the runner and start the configuration experience
$ ./config.sh --url https://github.com/stebje-actions-packages --token
AMVHBKYWH2WBXCI474ZW6DAX2RRI
# Last step, run it!
$ ./run.sh
```

The "Using your self-hosted runner" section includes a note: "# Use this YAML in your workflow file for each job runs-on: self-hosted". At the bottom, there's a link "For additional details about configuring, running, or shutting down the runner, please check out our [product docs](#)." A green button at the bottom right says "Back to runner settings".

# Runner groups

- Can be set up on enterprise and/or organization level
- Can be scoped to specific organizations and/or repositories
- Runners can be moved between groups
- A runner can only be in one group at a time

## Self-hosted runners

Host your own runners and customize the environment used to run jobs in your GitHub Actions workflows. Runners added to this organization can be used to process jobs in multiple repositories in your organization. [Learn more about self-hosted runners](#)

The screenshot shows the 'Runner groups' section of the GitHub interface. It displays a single group named 'Default' which covers 'All repositories'. There are 0 runners listed under this group. On the right side, there are three buttons: 'Add new', 'New runner', and 'New group'. The 'New group' button is highlighted in blue, indicating it is the active or selected action.

### Create group

#### Group name

macosx

#### Repository access: Selected repositories

##### ✓ Selected repositories

Runners can be used by specifically selected repositories

##### All repositories

Runners can be used by private and internal repositories

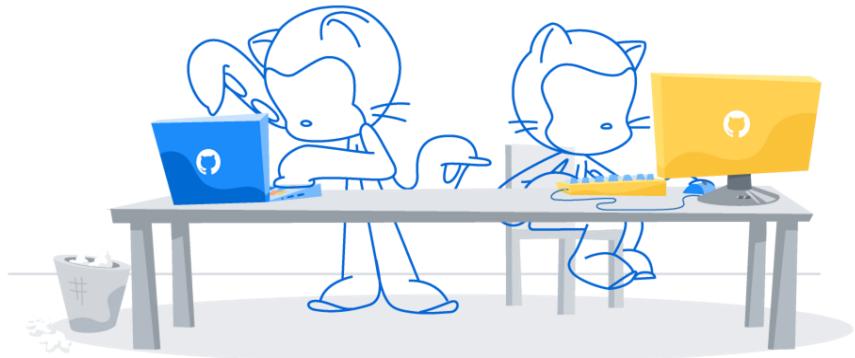
##### admin-tasks

# Security with self-hosted runners



Public repositories with self-hosted runners pose potential risks:

- Malicious programs running on the machine
- Escaping the machine's runner sandbox
- Exposing access to the machine's network
- Persisting unwanted or dangerous data on the machine



## Self-hosted runners and Security

Forked repositories will contain the same Actions configuration as the parent repository, including the self-hosted runners. Creates the potential for a fork to run malicious code on a runner inside your network. For this reason, it is highly recommended to use self-hosted runners only with **private** repositories.

# Scaling runners

- Auto-scaling is not yet supported with GitHub-hosted runners
- Open-source solutions do exist for scaling self-hosted runners, e.g.
  - <https://github.com/actions-runner-controller/actions-runner-controller>
  - <https://github.com/philiplabs/terraform-aws-github-runner>
- See <https://github.com/jonico/awesome-runners> for an open source list of options

jonico / awesome-runners

Watch 11 Unstar 110 Fork 11

Code Issues Pull requests Actions Projects Wiki Security

main Go to file Add file Code

jonico Merge pull request #2 from callum-tait-pbx/pa... 16 days ago 61

assets/css Update style.scss 3 months ago

LICENSE Apache 2 license 4 months ago

README.md Merge pull request #2 from callum-tait-pb... 16 days ago

\_config.yml Set theme jekyll-theme-hacker 4 months ago

About

A curated list of awesome self-hosted GitHub Action runners in a large comparison matrix

[jonico.github.io/awesome...](https://jonico.github.io/awesome-runners/)

github docker kubernetes aws security arm collection azure actions gcp self-hosted comparison operator awesome-list elastic auto-scaling scaling actions-runner comparisons-page self-hosted-actions

Readme Apache-2.0 License

README.md



awesome-runners

badges awesome License Apache-2.0 Made with Markdown Maintained? yes

Open Source? Yes!

Contributors 4

jonico Johannes Nicolai tetchel Tim Etchells MonolithProjects Mic...



Demo

# CI / CD workflows

# Basic CI workflow

- Uses a build matrix across multiple node versions
- Runs on the VM
  - Ubuntu in this case
- Actions are composable
  - Checkout is separate
  - Setup for most languages in [github.com/actions](https://github.com/actions)
  - npm run by shell
  - Artifact uploaded separately

```
name: Node CI

on: [push]

jobs:
  build:
    runs-on: ubuntu-latest

    strategy:
      matrix:
        node-version: [10.x, 12.x]

    steps:
      - uses: actions/checkout@v2
      - name: Use Node.js ${{ matrix.node-version }}
        uses: actions/setup-node@v2
        with:
          node-version: ${{ matrix.node-version }}
      - name: Install and test
        run: |
          npm ci
          npm run build --if-present
          npm test
      - uses: actions/upload-artifact@v2
        with:
          name: artifact
          path: dist/
```

# Linting

- Linting as part of CI runs
- See e.g. the super-linter
  - <https://github.com/github/super-linter>
  - Supports ~45 different languages
- Easily added as a new step to an existing workflow

```
name: Lint Code Base

on:
  push:
    branches-ignore: [main]
  pull_request:
    branches: [main]

jobs:
  build:
    name: Lint Code Base
    runs-on: ubuntu-latest
    steps:
      - name: Checkout Code
        uses: actions/checkout@v2
        with:
          fetch-depth: 0

      - name: Lint Code Base
        uses: github/super-linter@v4
        env:
          VALIDATE_ALL_CODEBASE: false
          DEFAULT_BRANCH: main
          GITHUB_TOKEN:
            ${{ secrets.GITHUB_TOKEN }}
```

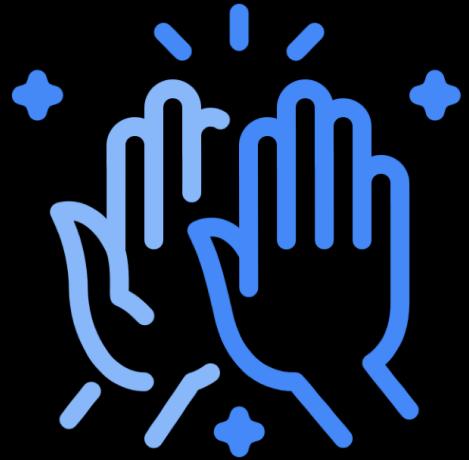
# Basic CD workflow

- Starter workflows available for most cloud providers
- Store the image in GitHub
- Jobs run on different envs
  - Uses the Docker image
  - Deploys the container image to Azure

```
35 Build-Docker-Image:  
36   runs-on: ubuntu-latest  
37   needs: build  
38   name: Build image and store in GitHub Packages  
39   steps:  
40     - name: Checkout  
41       uses: actions/checkout@v1  
42  
43     - name: Download built artifact  
44       uses: actions/download-artifact@master  
45       with:  
46         name: webpack artifacts  
47         path: public  
48  
49     - name: create image and store in Packages  
50       uses: mattdavis0351/actions/docker-gpr@1.3.0  
51       with:  
52         repo-token: ${{secrets.GITHUB_TOKEN}}  
53         image-name: ${{env.DOCKER_IMAGE_NAME}}  
54  
55 Deploy-to-Azure:  
56   runs-on: ubuntu-latest  
57   needs: Build-Docker-Image  
58   name: Deploy app container to Azure  
59   steps:  
60     - name: "Login via Azure CLI"  
61       uses: azure/login@v1  
62       with:  
63         creds: ${ secrets.AZURE_CREDENTIALS }  
64  
65     - uses: azure/docker-login@v1  
66       with:  
67         login-server: ${{env.IMAGE_REGISTRY_URL}}  
68         username: ${{ github.actor }}  
69         password: ${{ secrets.GITHUB_TOKEN }}  
70  
71     - name: Deploy web app container  
72       uses: azure/webapps-container-deploy@v1  
73       with:  
74         app-name: ${{env.AZURE_WEBAPP_NAME}}  
75         images: ${{env.IMAGE_REGISTRY_URL}}/${{ github.repository }}/${{env.DOCKER_IMAGE_NAME}}:${{ github.sha }}  
76  
77     - name: Azure logout  
78       run: |  
79         az logout
```



Demo



Q&A



Thank you