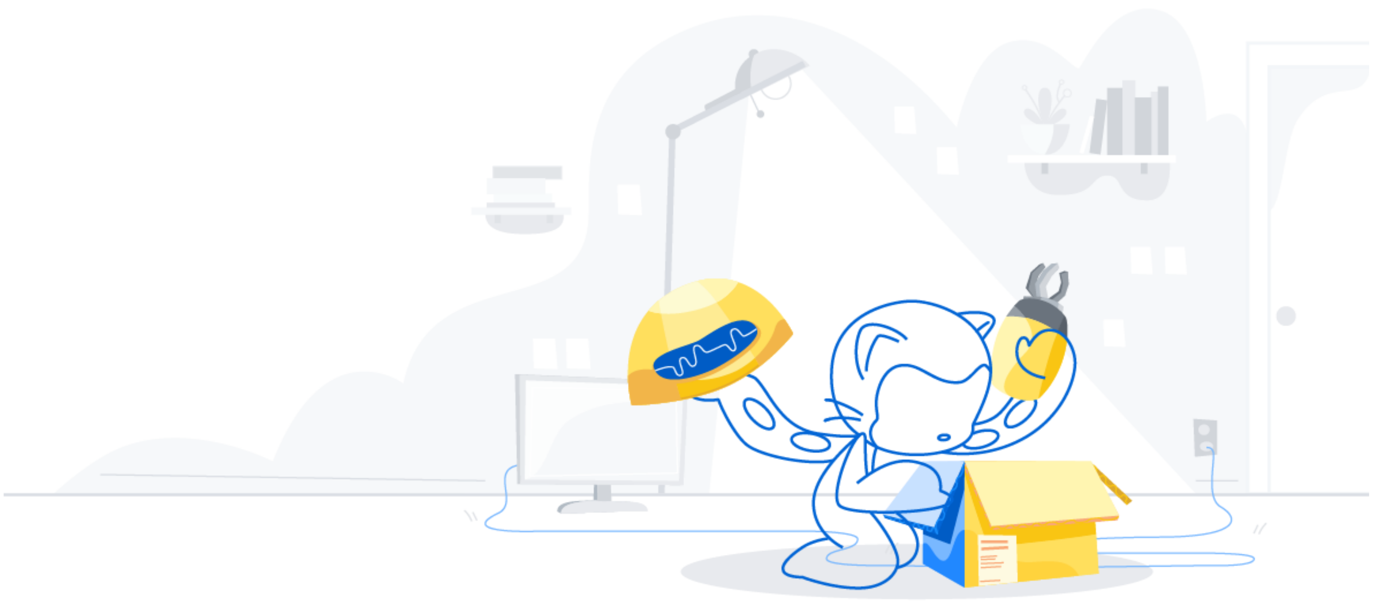
7 advanced workflow automation features with GitHub Actions

Check out some advanced automation and CI/CD capabilities you can use today with GitHub Actions on any GitHub account.



Author

Brian Douglas

November 18, 2021

TL;DR: Check out some advanced automation and CI/CD capabilities you can use today with GitHub Actions on any GitHub account.

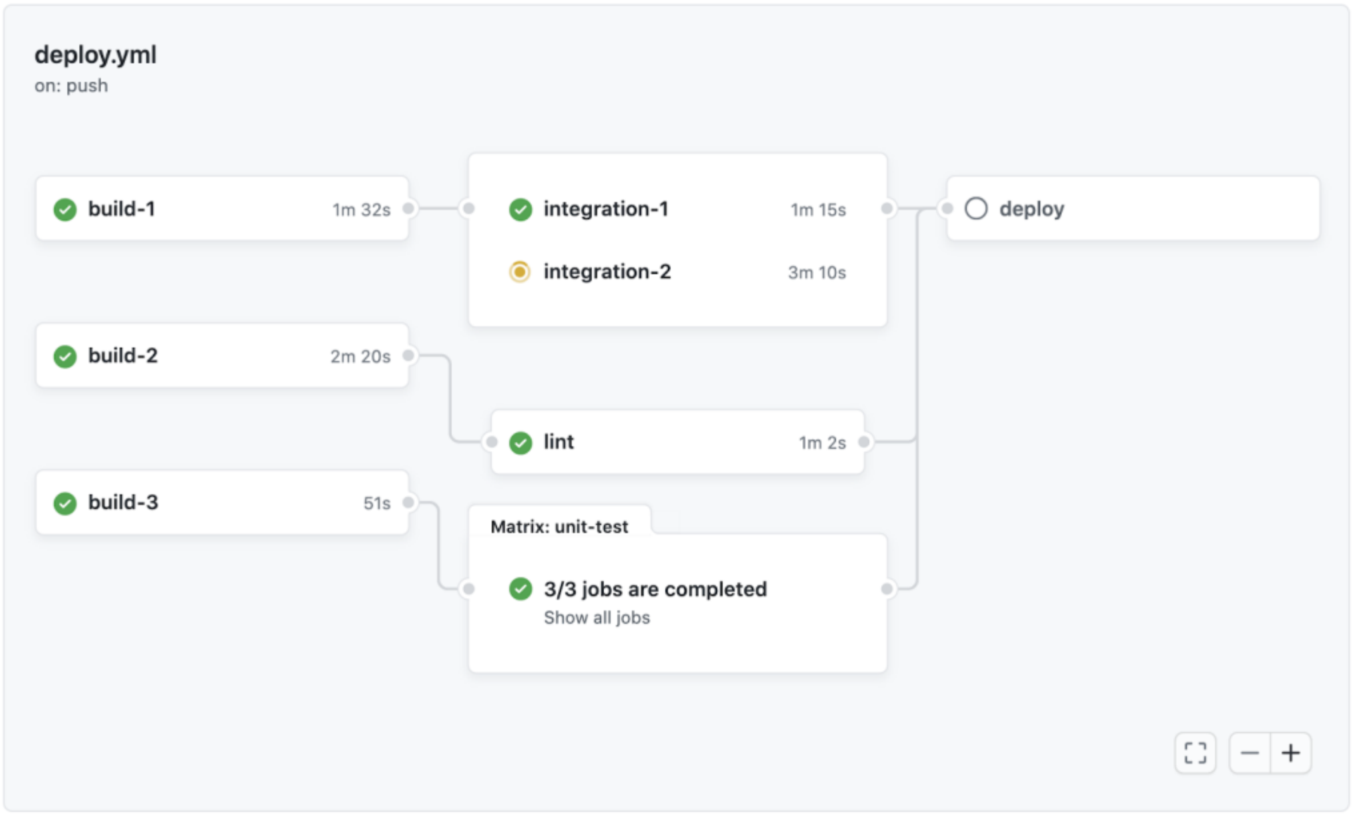
GitHub Actions is designed to bring platform-native automation and CI/CD capabilities directly into the GitHub flow to simplify the developer experience. It can also be used to build out more advanced, custom workflows for anything from triggering an alarm to orchestrating complex security test automations.

If you're looking to create advanced workflows, you can use the pre-built actions in the GitHub Actions Marketplace (we just passed 10,000!) to help.

But if you're creating your own custom workflows, here are seven features to keep in mind (you can also learn more about managing complex workflows with GitHub Actions in GitHub Docs, or watch my on demand session at GitHub Universe).

1. Track your workflows with workflow visualization

All actions start and end with YAML files—but you can also access real-time workflow visualization graphs to track progress, understand dependencies and conditionals in more complex workflows, and troubleshoot any issues that come up via logs. Plus, workflow visualization graphics are color-coded to quickly show you what actions were successful, which are in progress, and which failed at a particular step.



2. Build breakpoints in a workflow with dependencies (or create dependencies between workflows)

GitHub Actions runs multiple commands at the same time by default—but you can leverage the `needs` keyword to create dependencies between jobs. That means that if something like a test fails (or any job for that matter) dependent jobs won't execute.

You can also create dependencies between workflows. This can create connection points—and breakpoints, for that matter—between automations. (Pro tip: You can cache dependencies to speed up workflows and reduce run times.)

3. A conditional can make all the difference

GitHub Actions supports conditionals, which use the `if` keyword to determine if a step should run in a given workflow. You can use this to build upon dependencies so that if a dependent job fails the workflow can continue running. These can use certain built-in functions for data operations.

You can also leverage status check functions to determine if previous steps have succeeded, failed, been canceled, or otherwise disrupted. Another useful benefit of conditionals: you can use them to share workflow data between different branches and forks with steps tailored to different triggers or environments.

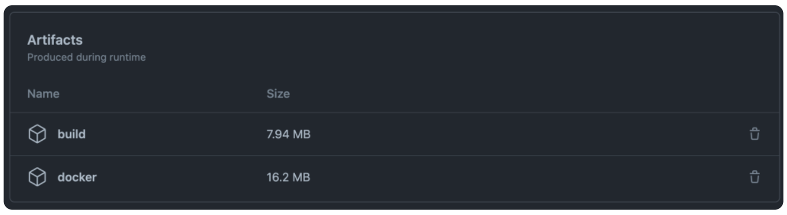
4. Creating workflows with sensitive data? Try storing secrets

Secrets on GitHub are used to securely store sensitive data such as passwords, tokens, and certificates, among other things—and they can be directly referenced in workflows, too. This means you can create and share workflows with collaborators using secrets for secure values without hardcoding those values directly in YAML files.

```yaml
name: PR text MSG on failure
on: [pull_request]
jobs:
  build:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v2
      - name: Run a one-line script
        run: npm run build

  send-a-text-text:
    if: ${{ failure() }}
    needs: build
    runs-on: ubuntu-latest
    steps:
    - name: 'Sending SMS Notification'
      uses: twilio-labs/actions-sms@v1
      with:
        fromPhoneNumber: '+1(267)8282212'
        toPhoneNumber: ${{ secrets.PAGER_NUMBER }}
        message: 'The build failed'
      env:
        TWILIO_ACCOUNT_SID: ${{ secrets.TWILIO_ACCOUNT_SID }}
        TWILIO_API_KEY: ${{ secrets.TWILIO_API_KEY }}
        TWILIO_API_SECRET: ${{ secrets.TWILIO_API_SECRET }}
```

An example of a conditional in a GitHub Actions workflow.



Artifacts
Produced during runtime

Name	Size
build	7.94 MB
docker	16.2 MB

An example of artifacts in GitHub Actions that are used to share data between jobs in an automation workflow

5. Share data between jobs to create more advanced automations

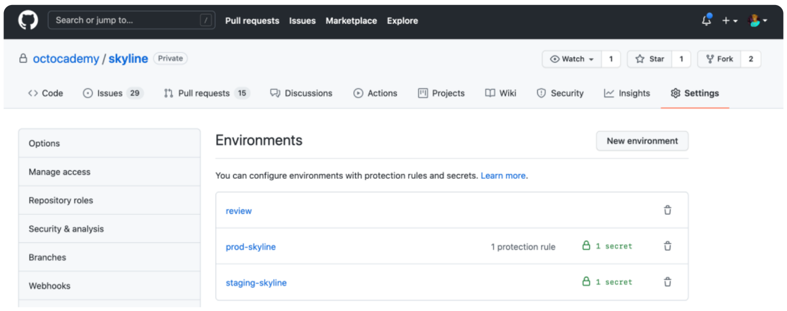
GitHub Actions supports sharing data between jobs in any workflow as artifacts, which are associated with the workflow run where they are created. This can help simplify the development of YAML workflow files. It also makes it easier to create more complex automations where one workflow informs another via dependencies or conditionals.

6. Use contexts to access workflow information

Contexts are a collection of variables used to access information about workflow runs, runner environments, jobs and steps to help derive key information about workflow operations. Contexts use expression syntax like `${{ }}`. Most of them can be used at any point in the workflow.

7. Create different environments for development, staging, and production

GitHub Actions allows you to create environments with secure protection rules and secrets—and then reference those environments in a workflow job to leverage the environment's protection rules and secrets. Example use cases include requiring a specific person or team to approve workflow jobs that reference an environment or restricting which branches can deploy to a given environment.



Environments

You can configure environments with protection rules and secrets. Learn more.

review		
prod-skyline	1 protection rule	1 secret
staging-skyline		1 secret

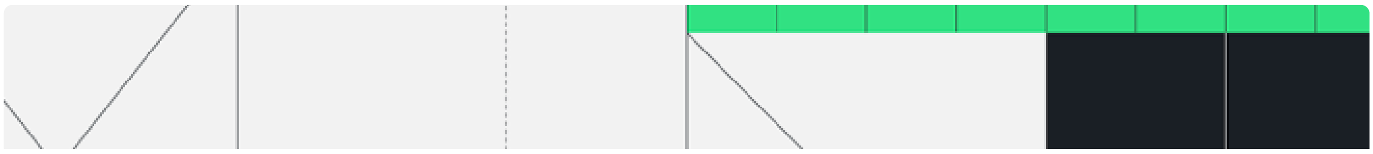
An example of different environments in GitHub Actions workflow automations.

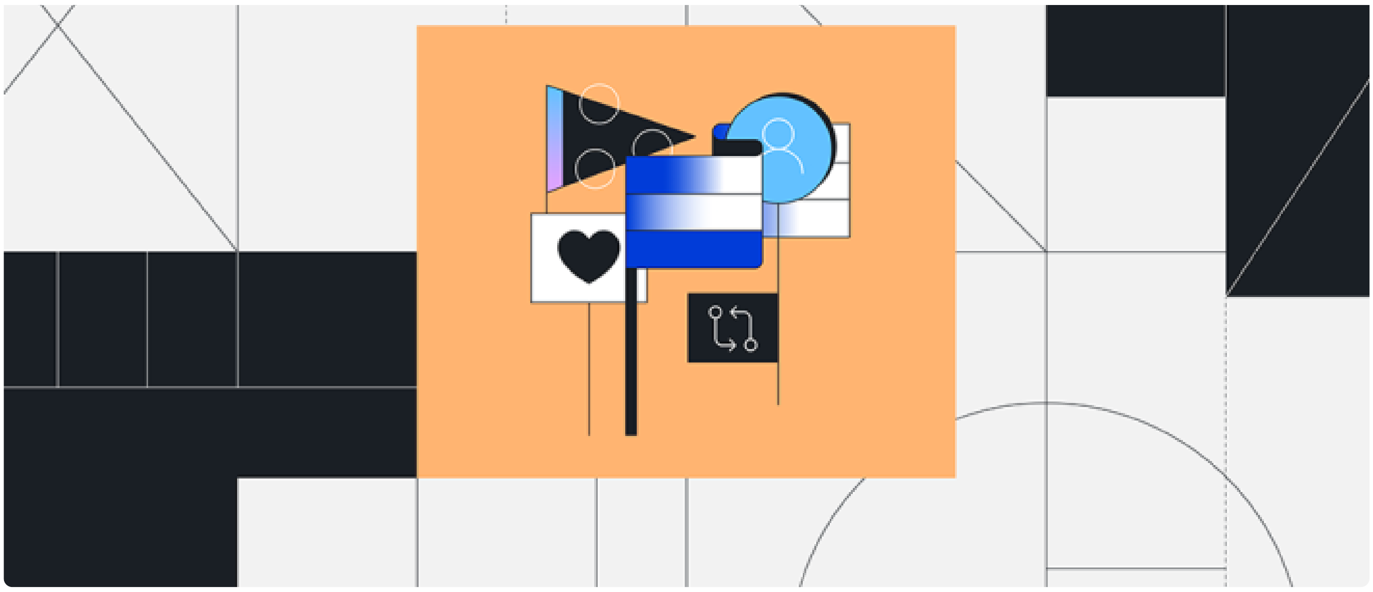
Have questions?

Try our Docs pages for more insights on what GitHub Actions is and can do for you.

Tags: automation, DevOps, GitHub Actions

More on **automation**





5 automations every developer should be running

Looking to avoid security vulnerabilities, buttons that don't work, slow site speeds, or manually writing release notes? This one's for you.

Brian Douglas

---

5 DevOps tips to speed up your developer workflow

From learning YAML to scripting with Bash, here are a few simple tips for developers who want to speed up their workflows.

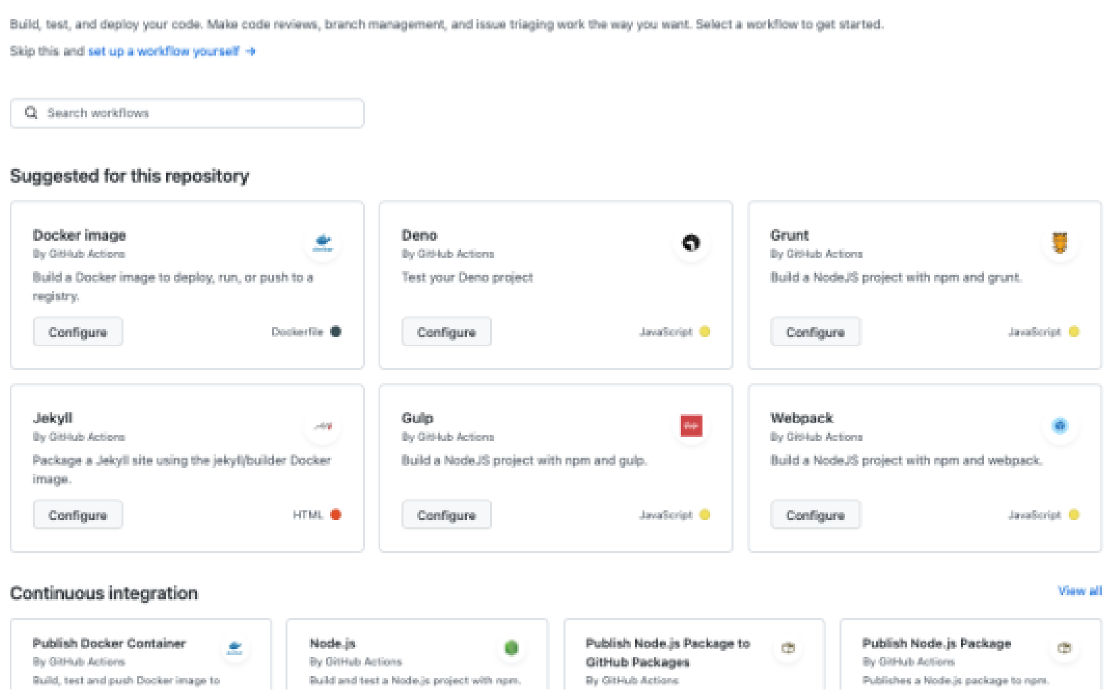
Damian Brady

Getting started with DevOps automation

This is the second post in our series on DevOps fundamentals. For a guide to what DevOps is and answers to common DevOps myths check out part one. What role...

Jared Murrell

More on **DevOps**



Build, test, and deploy your code. Make code reviews, branch management, and issue triaging work the way you want. Select a workflow to get started.

Skip this and set up a workflow yourself →

Search workflows

Suggested for this repository

Docker image
By GitHub Actions
Build a Docker image to deploy, run, or push to a registry.
Configure Dockerfile

Deno
By GitHub Actions
Test your Deno project
Configure JavaScript

Grunt
By GitHub Actions
Build a NodeJS project with npm and grunt.
Configure JavaScript

Jekyll
By GitHub Actions
Package a Jekyll site using the jekyll/builder Docker image.
Configure HTML

Gulp
By GitHub Actions
Build a NodeJS project with npm and gulp.
Configure JavaScript

Webpack
By GitHub Actions
Build a NodeJS project with npm and webpack.
Configure JavaScript

Continuous integration View all

Publish Docker Container
By GitHub Actions
Build, test and push Docker image to

Node.js
By GitHub Actions
Build and test a Node.js project with npm.

Publish Node.js Package to GitHub Packages
By GitHub Actions

Publish Node.js Package
By GitHub Actions
Publishes a Node.js package to npm.

How to build a CI/CD pipeline with GitHub Actions in four simple steps

A quick guide on the advantages of using GitHub Actions as your preferred CI/CD tool—and how to build a CI/CD pipeline with it.

Brian Douglas

5 DevOps tips to speed up your developer workflow

From learning YAML to scripting with Bash, here are a few simple tips for developers who want to speed up their workflows.

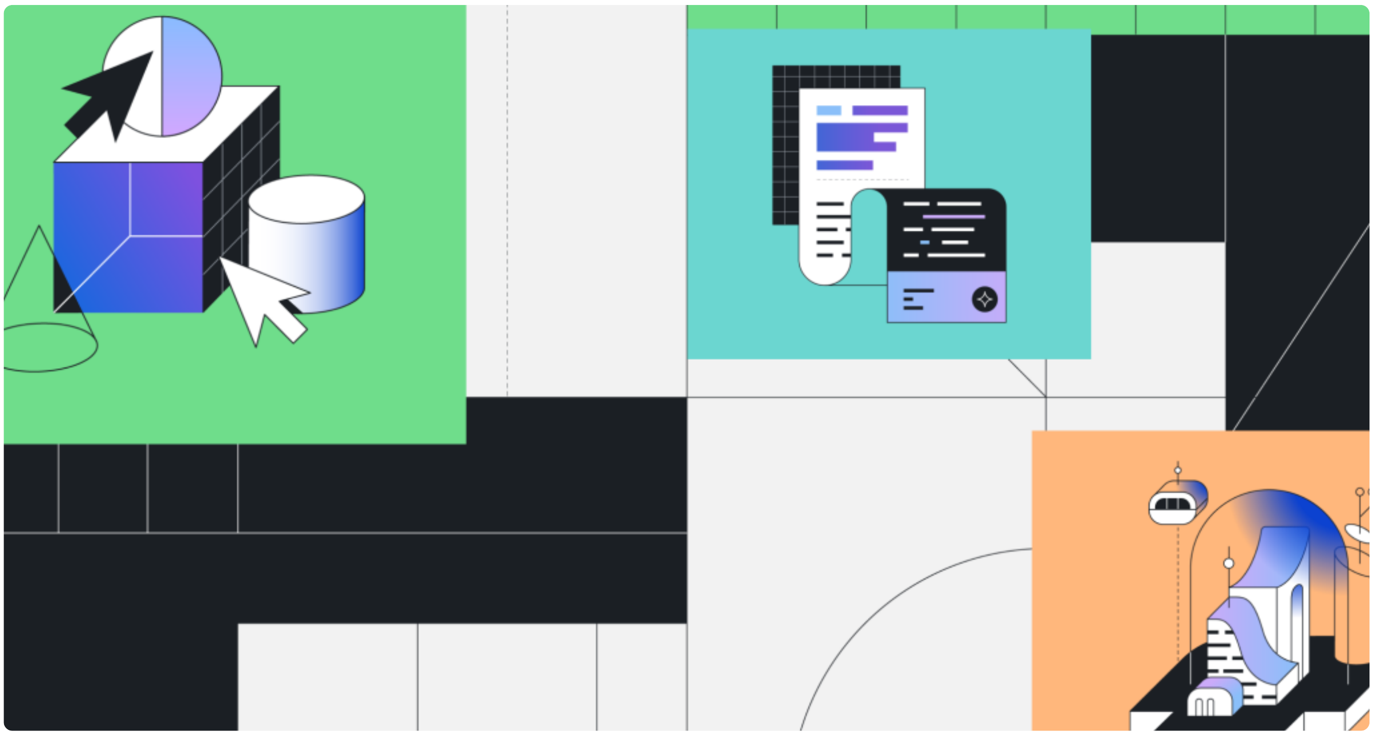
Damian Brady

10 GitHub Actions resources to bookmark from the basics to CI/CD

Tips on how to get started using GitHub Actions and resources to learn more about making it work for you.

Brian Douglas

Related posts



What's new in Codespaces for Organizations

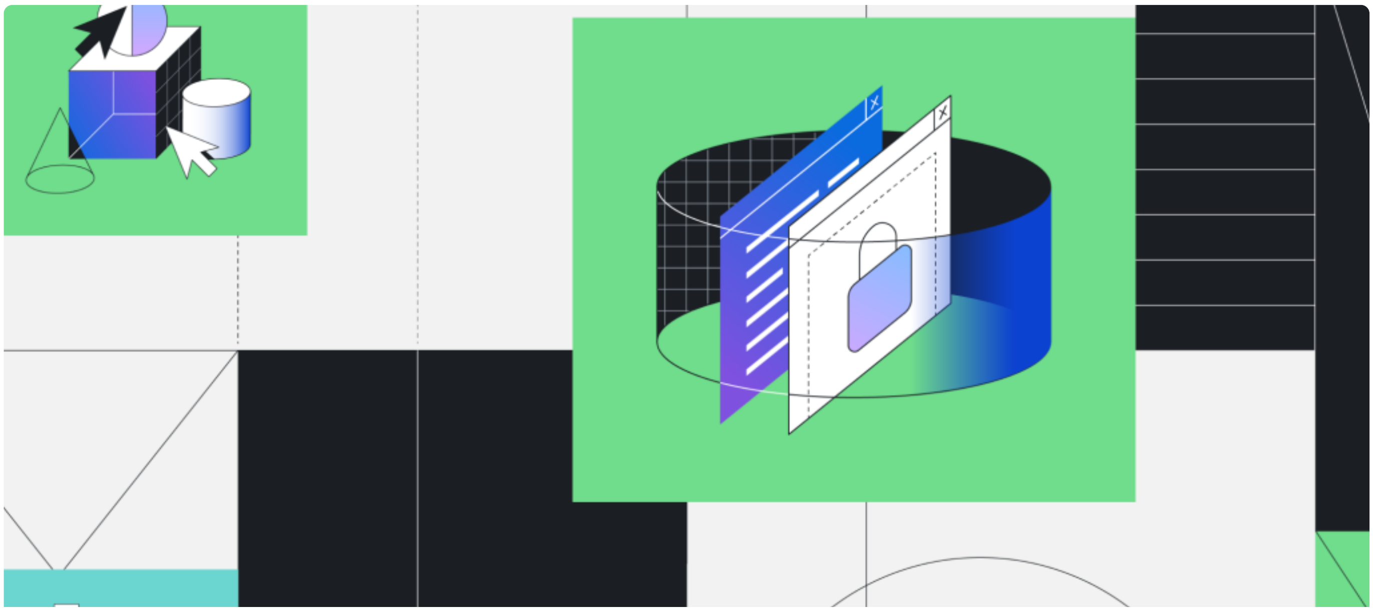
We're releasing exciting functionalities that will enable organizations to confidently manage and scale with Codespaces.

Tanmayee Kamath

GitHub Copilot is generally available to all developers

We're making GitHub Copilot, an AI pair programmer that suggests code in your editor, generally available to all developers for $10 USD/month or $100 USD/year. It will also be free to use for verified students and maintainers of popular open source projects.

Thomas Dohmke



Creating a more comprehensive dependency graph with build time detection

Expand the completeness of your dependency graph by using the dependency submission API, which will create more comprehensive alerts on supply chain vulnerabilities

Courtney Claessens

Explore more from GitHub

Product

Updates on GitHub products and features, hot off the press.

Learn more >

The ReadME Project

Stories and voices from the developer community.

Learn more

GitHub Actions

Native CI/CD alongside code hosted in GitHub.

Learn more

Work at GitHub!

Check out our current job openings.

Learn more

Subscribe to The GitHub Insider

A newsletter for developers covering techniques, technical guides, and the latest product innovations coming from GitHub.

Your email address