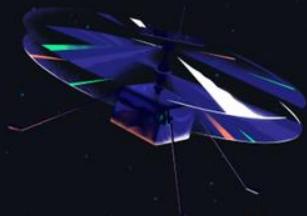




# GitHub Training

3/25/25



## AGENDA

GitHub Codespaces

GitHub Actions

GitHub Advanced Security

GitHub Copilot



# GitHub Platform



# GitHub Codespaces



# Modern cloud development

Improving your organization's software  
development process  
with Codespaces



# Impact of local development and expanding engineering teams



## Onboarding friction

New engineers spending more time figuring out how to set up locally rather than understanding the codebase.



## Over or under investment in developer experience

Daily tools for your engineering teams should *just work*.



---

## Opportunities

Improving engineer  
satisfaction and  
productivity

“

Teams continue to move workloads to the cloud and those that leverage all five capabilities of cloud see increases in software delivery and operational (SDO) performance, and in organizational performance.

Accelerate State of Devops Report 2021

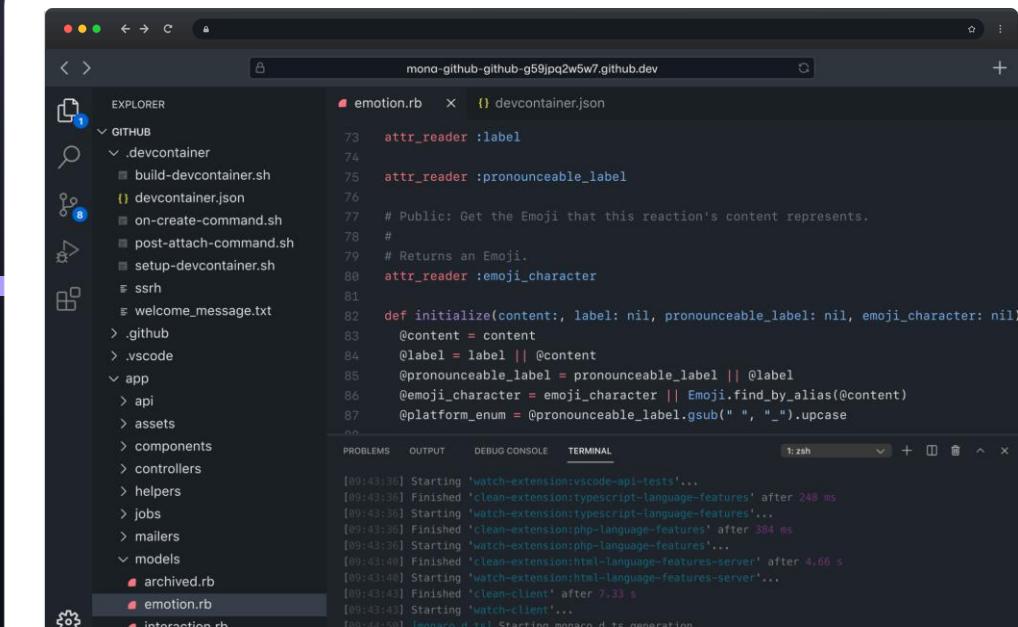


GITHUB CODESPACES

# GitHub Codespaces

## Blazing fast cloud developer environments

Full compute backing, up to 32 cores and 64 GB RAM



A screenshot of a GitHub Codespace interface. On the left, the Explorer sidebar shows a file tree with a .github folder containing .devcontainer, build-devcontainer.sh, devcontainer.json, on-create-command.sh, post-attach-command.sh, setup-devcontainer.sh, and ssrh. It also includes welcome\_message.txt, .github, .vscode, app, api, assets, components, controllers, helpers, jobs, mailers, models (with archived.rb, emotion.rb, and interaction.rb selected), and a .gitignore file. On the right, the main area has tabs for emotion.rb and devcontainer.json. The emotion.rb tab displays a code editor with a snippet of Ruby code. Below the code editor are tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL. The TERMINAL tab shows a log of command-line activity, including logs for 'watch-extension:vscode-api-tests', 'clean-extension:typescript-language-features', 'watch-extension:typescript-language-features', 'clean-extension:php-language-features', 'watch-extension:php-language-features', 'clean-extension:html-language-features-server', 'watch-extension:html-language-features-server', 'clean-client', and 'watch-client'. A browser window in the background shows a 10x10 grid of small gray dots.

```
attr_reader :label
attr_reader :pronounceable_label
# Public: Get the Emoji that this reaction's content represents.
# Returns an Emoji.
attr_reader :emoji_character
def initialize(content:, label: nil, pronounceable_label: nil, emoji_character: nil)
  @content = content
  @label = label || @content
  @pronounceable_label = pronounceable_label || @label
  @emoji_character = emoji_character || Emoji.find_by_alias(@content)
  @platform_enum = (@pronounceable_label.gsub(" ", "_").upcase
  ...
[09:43:36] Starting 'watch-extension:vscode-api-tests'...
[09:43:36] Finished 'clean-extension:typescript-language-features' after 248 ms
[09:43:36] Starting 'watch-extension:typescript-language-features'...
[09:43:36] Finished 'clean-extension:php-language-features' after 384 ms
[09:43:36] Starting 'watch-extension:php-language-features'...
[09:43:40] Finished 'clean-extension:html-language-features-server' after 4.66 s
[09:43:40] Starting 'watch-extension:html-language-features-server'...
[09:43:40] Finished 'clean-client' after 7.33 s
[09:43:40] Starting 'watch-client'...
fan.vscs1 [main] Starting workspace_d.ts generation
```



## Mean time to onboard

Faster bootstrap with a cloud-native experience

## Improved auditing

Better visibility and traceability over local development

## Standardized environments

Customize environments per repository, standardizing the development experience



# Codespaces Lab



# Create Node App and Test Coverage

- Create a new GH repo called **node-coverage-lab**
- Launch new Codespace on main branch(green button)
- When Codespace loads, run the following commands:
- **npm init -y**
- **npm install --save-dev jest**
- Verify node version: **node --version**
- Create new file **app.js**(add supplied code)
- Create a new folder at the root called **test**
- Create a new file **app.test.js**(add supplied code)
- Run git add ., git commit -m “ ”, git push origin main
- Update package.json file to use Jest test case
- Run npm test



# Update .devcontainer configuration

- At the root, create a new folder called **.devcontainer**
- Create a new file named devcontainer.json
- Add supplied configuration file
- Open command prompt(crtl+shift+p)
- Select “Codespaces: Rebuild Container”
- Verify Node version: node --version



# GitHub Actions



# GitHub Actions Fundamentals



Presented by GitHub Professional Services



# Our Agenda

01

## GitHub Actions

Introduction

02

## Actions Workflow

Syntax

03

## Actions Secrets

Environments and secrets

04

## Manage Actions

05

## Building Actions

Build Actions & Workflows

01

## Migration

Migrate to Actions

02

## Runners

GitHub Actions Runners

03

## CI/CD

Action as CI/CD workflows

04

## Labs & Demos

Actions Demos



# GitHub Actions

## Introduction



## What's GitHub Actions

**GitHub Actions is a core CI/CD & automation feature that operates natively within an integrated DevOps platform**

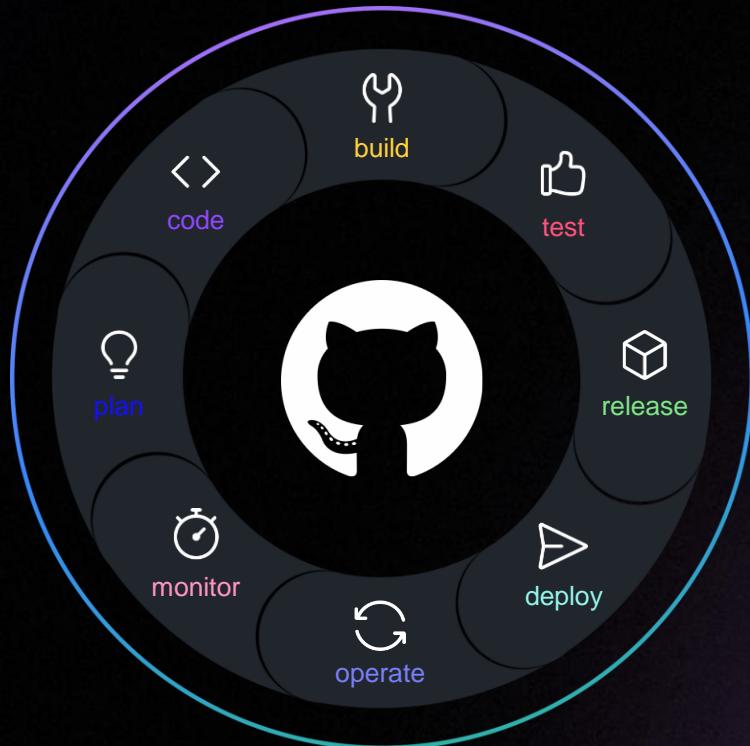
- ✓ Workflows stored as yml files
- ✓ Fully integrated with GitHub
- ✓ Respond to GitHub events
- ✓ Live logs and visualized workflow execution
- ✓ Community-powered and custom workflows
- ✓ GitHub-hosted, self-hosted, or Kubernetes hosted runners
- ✓ Built-in secrets and environment variables store



# Actions & SDLC

## Use cases across SDLC using GitHub Actions

- Plan**  
Tirage Issues and Project Boards using Actions
- Code**  
Automates code builds, linting upon pull request creation
- Build**  
Compile and build, Store artifacts
- Test**  
Automation of the unit tests, integration tests
- Release**  
Publish release, Release notes, Versioning, and Upload artifacts
- Deploy**  
Deploy applications to your chosen environment





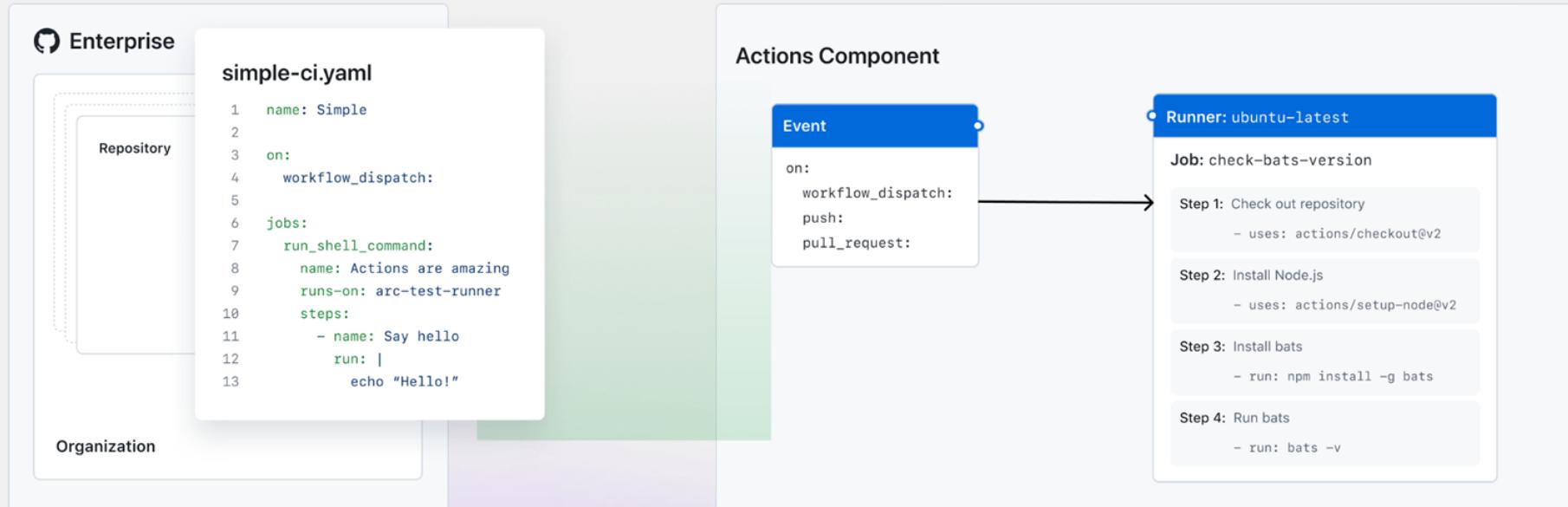
# Automation accessible to every developer

The screenshot shows the GitHub Actions interface. At the top, there's a blue header with the GitHub logo and the word "Actions". Below it, a green title "Actions" is displayed next to a blue icon of three interconnected nodes. A sub-section below the title says "Enterprise-grade CI/CD that supports Windows, Linux, Mac". The main area shows a workflow named "create\_demo.yml" triggered by "workflow\_dispatch". The status bar indicates it was "Manually triggered 2 minutes ago" by "octodemobot" and is currently "In progress". The workflow has four steps: "Initialize" (green), "Configure Creation Steps" (green), "Create Demo Repository" (yellow), and "Deploying to octodemo/testing-provisioning" (blue). The "Create Demo Repository" step is currently running and has a duration of "1m 12s". To the right of the workflow, there are several options: "Register Demo Deployment S...", "Update tracking issue", "Matrix: Configure secrets", and "Waiting for pending jobs". Below the workflow, there are tabs for Discussions, Actions (which is active), Projects, Wiki, Security, Insights, and Settings.

- ✓ Marketplace of 20,000+ Actions by our community
- ✓ Any operating system, cloud and on-prem
- ✓ Natively integrated into GitHub workflows
- ✓ Friction-free service or self-hosted runners
- ✓ Integrates Azure Pipelines & any other CI/CD
- ✓ DRY with Reusable Workflows
- ✓ API for viewing cache usage



# GitHub Actions Key Components





# Actions Workflow Syntax



# Understanding the workflow file

The **name of the workflow** as it will appear in the "Actions" tab of the GitHub repository

The **name for workflow runs** generated from the workflow, which will appear in the list of workflow runs on your repository's "Actions" tab

Specifies **the trigger for this workflow**.

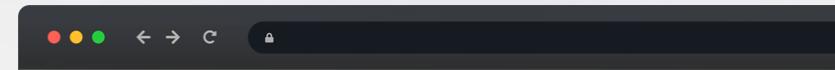
This example uses the push event, so a workflow run is triggered every time someone pushes a change to the repository or merges a pull request

Groups together all the **jobs** that run in the learn-github-actions workflow.

Defines a job named **check-bats-version**. The child keys will define properties of the job.

Configures the **job runner** on the latest version of an Ubuntu Linux runner.

Groups together all the **steps** that run in the **check-bats-version** job and the `uses` keyword specifies that this step will run **v4** of the **actions/checkout** action



```
name: learn-github-actions

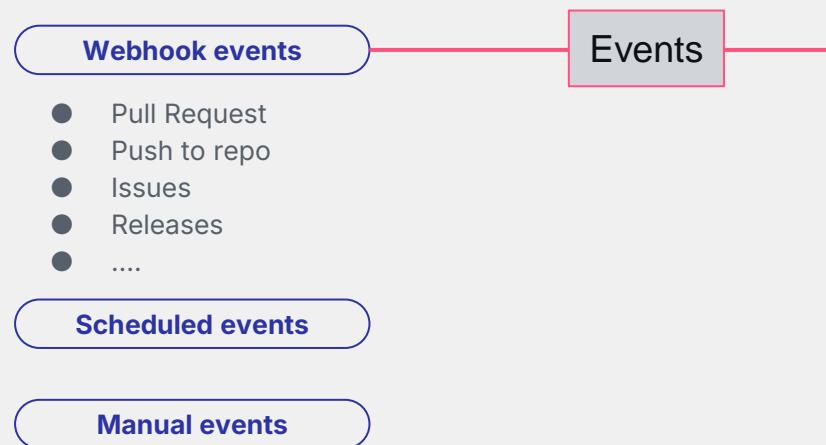
run-name: ${{ github.actor }} is learning GitHub Actions

on: [push]

jobs:
  check-bats-version:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v4
```



# Events

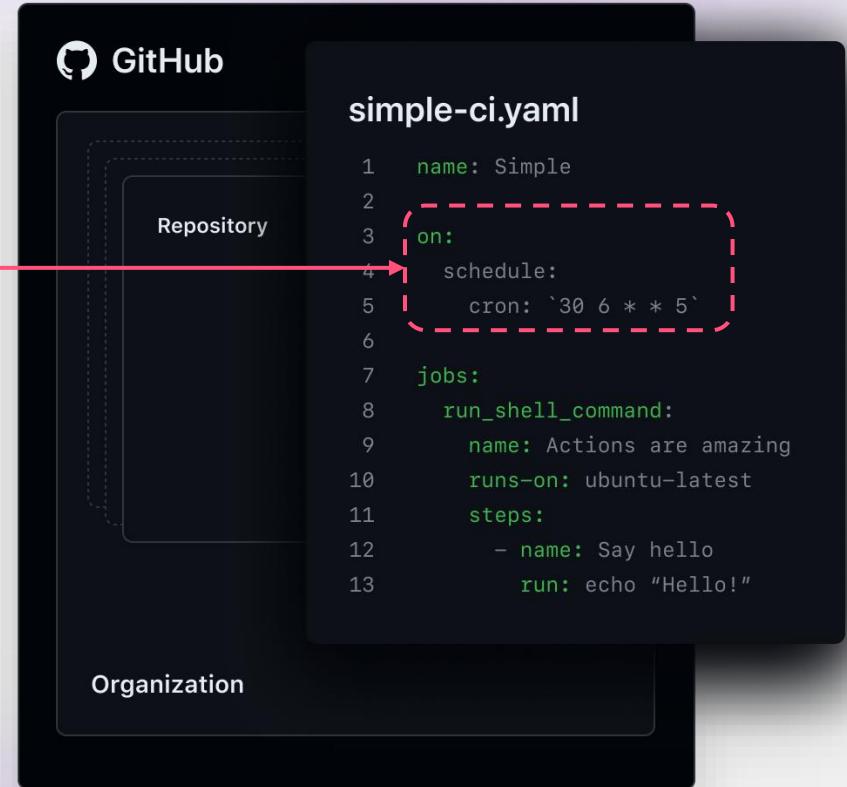
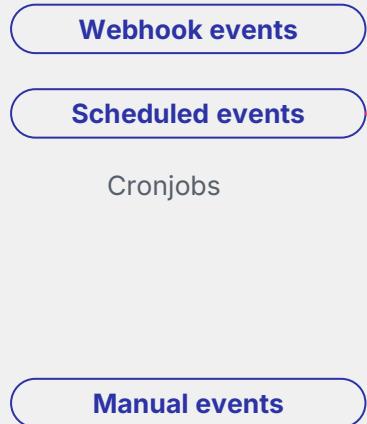


The screenshot shows a GitHub interface with a repository named "simple-ci". A callout box highlights the "simple-ci.yaml" file, which contains the following YAML configuration:

```
1 name: Simple
2
3 on:
4   push
5
6 jobs:
7   run_shell_command:
8     name: Actions are amazing
9     runs-on: ubuntu-latest
10    steps:
11      - name: Say hello
12        run: echo "Hello!"
```



# Events





# Events

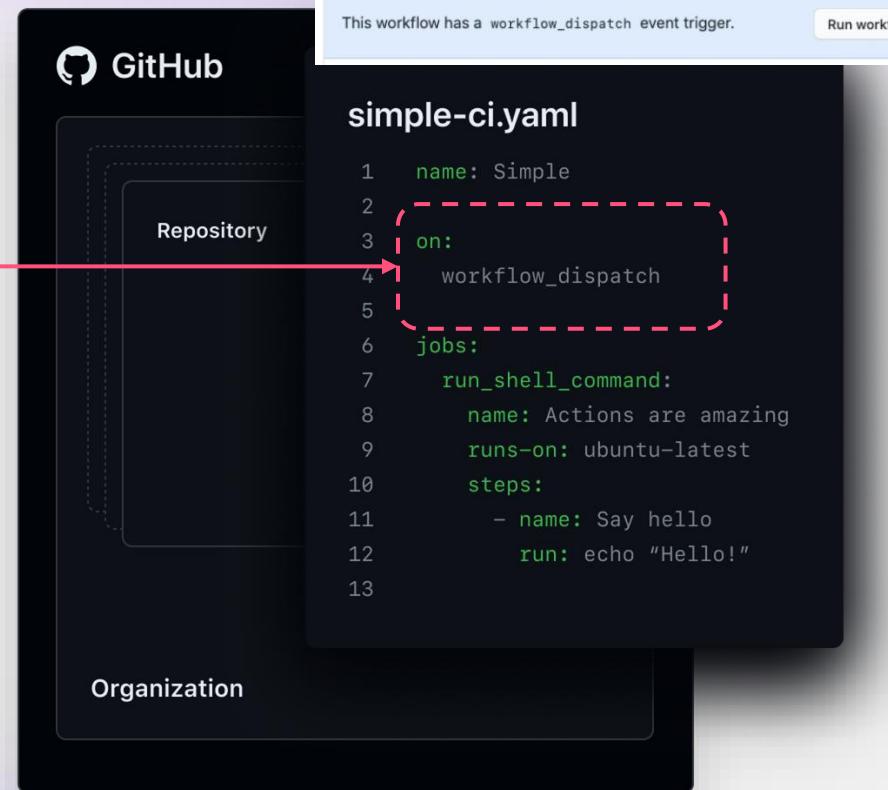
Webhook events

Scheduled events

Manual events

- workflow\_dispatch
- repository\_dispatch

Events





# Runners



GitHub Hosted Runners



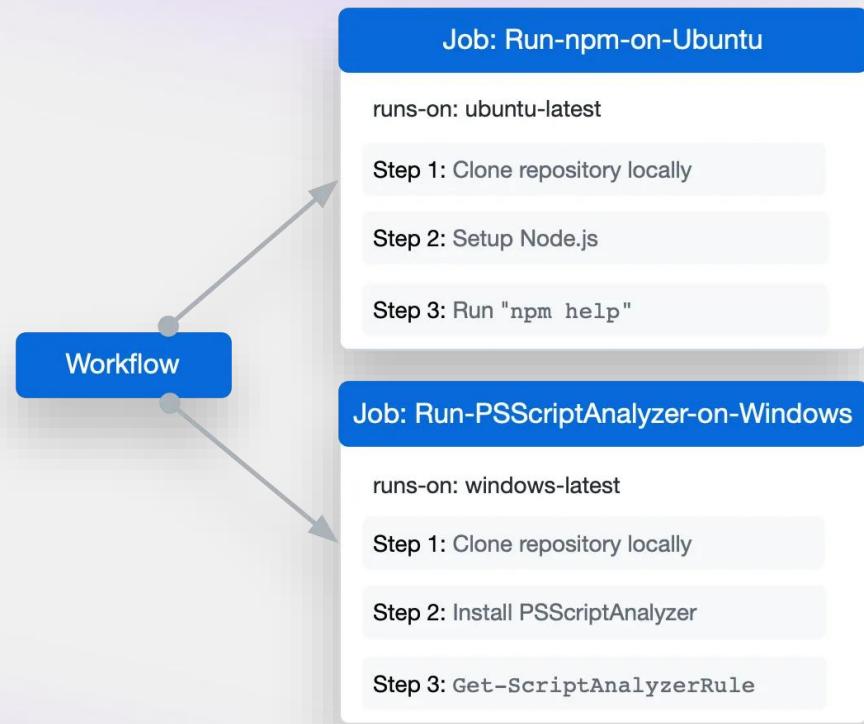
Self-Hosted Runners





# GitHub Hosted Runners

- Automatic Provisioning and Maintenance
- Diverse Operating System Support
- Integrated Software and Tools
- Scalability and Workflow Continuity
- Security and Administrative Privileges
- Customization and Local Environment Variables
- Using private vNet on Azure





# Self-Hosted Runners

- Full Control and Customization
- Flexibility in Hosting
- Cost Management
- Run on OS not supported on GitHub-hosted runner
- Usage Limits and Continuity
- Custom hardware config
- Can be grouped together

## Runner groups

Control access to your runners by specifying the repositories that are able to use your shared organization runners.

New runner group

Group	Runners	...
<b>Default</b> ⓘ All repositories, excluding public repositories	100	
<b>azure</b> Selected repositories (18), excluding public repositories	260	...
<b>ui-tests</b> Selected repositories (3), excluding public repositories	92	...
<b>mobile</b> Selected repositories (1), excluding public repositories	51	...
<b>aws</b> Selected repositories (10), excluding public repositories	104	...
<b>dotcom</b> Selected repositories (15), excluding public repositories	187	...

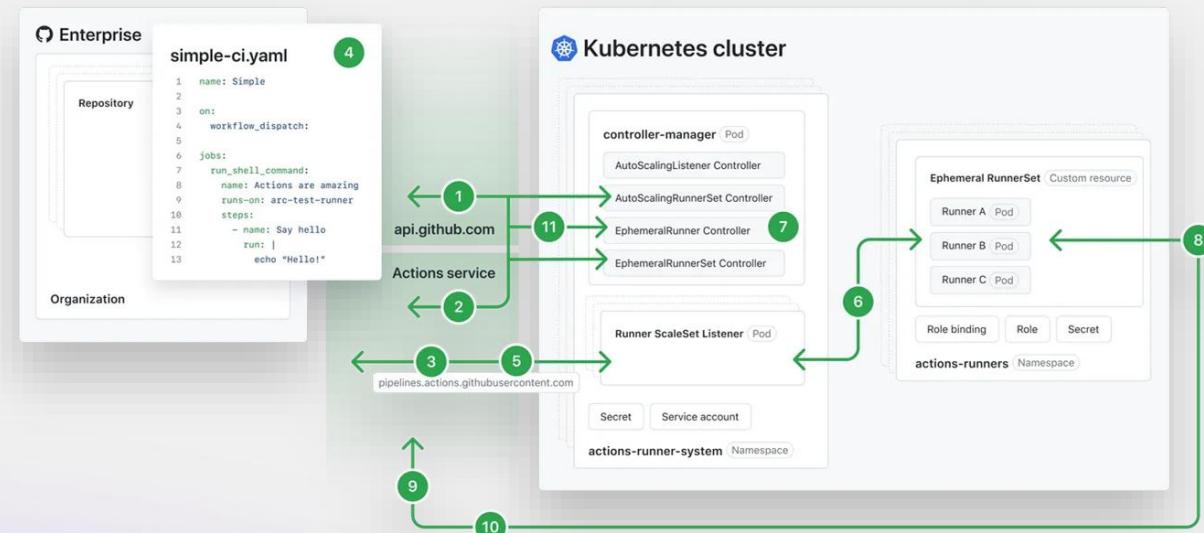
## Shared by the Enterprise

Group	Runners	...
<b>Default</b> All repositories, excluding public repositories	230	
<b>enterprise-hourly</b> All repositories, excluding public repositories	89	
<b>kubernetes</b> All repositories, excluding public repositories	452	



# Self-Hosted Runners - ARC

- Kubernetes Integration
- Autoscaling Capabilities
- Ephemeral Runners
- Container-Based Runners
- Customizable Installation
- Runner Container Image





# Actions

- Reusable units of code that can be referenced in a workflow
- GitHub runs them in Node.js runtime, or in containers
- Reference an Action, or run scripts directly

The diagram illustrates the scope of GitHub Actions. It shows three nested levels: **Enterprise** (the outermost), **Organization** (the middle), and **Repository** (the innermost). Dashed lines indicate the boundaries between these levels.

```
simple-ci.yaml
1 name: Simple
2
3 on:
4   push
5
6 jobs:
7   run_shell_command:
8     name: Actions are amazing
9     runs-on: ubuntu-latest
10    steps:
11      - name: Say hello
12        run: echo "Hello!"
13      - name: Public Action
14        uses: actions/checkout@v4
15      - name: Local Action
16        uses: ./path/to/action
17      - name: Docker Image
18        uses: docker://alpine:3.8
```

[Script](#)[Public Action](#)[Local Action](#)[Docker Image](#)



# Largest Connected Developer Community

“

What I love about GitHub Actions is that you're not limited in your choice of tools.

The screenshot shows a web browser window for GitHub.com displaying the Marketplace. The search bar at the top contains the query "sort:popularity-desc". Below the search bar, there are filter options: "Types" (with "Actions" selected), "Actions", and "Stacks". On the left, there's a sidebar titled "Categories" listing various tools: API management, Chat, Code quality, Code review, Continuous integration, Dependency management, Deployment, IDEs, Learning, Localization, and Mobile. On the right, the results are displayed under the heading "Actions". The first result is "TruffleHog OSS" by trufflesecurity, described as a tool for scanning GitHub Actions with TruffleHog. The second result is "Super-Linter" by github, described as a tool for validating source code. The third result is "GitHub Pages action" by peaceiris, described as a tool for GitHub Pages. A large blue banner at the bottom right of the page states "20k+ Actions available in the GitHub Marketplace".



# Starter workflows



Preconfigured for specific languages and frameworks



GitHub analyzes your code and suggests the workflows  
based on your language and framework



You can also create your own starter workflow to share with  
your organization.

The screenshot shows a web browser window with the GitHub URL in the address bar. The main content is titled "Choose a workflow" with the sub-instruction "Build, test, and deploy your code. Make code reviews, branch management, and issue triaging work the way you want. Select a workflow below." Below this, there's a link "Skip this and set up a workflow yourself →".

On the left, there's a sidebar with "Categories" (Automation, Continuous integration, Deployment, Security) and a search bar "Search workflows".

The main area displays "Found 37 workflows" with cards for several pre-configured workflows:

- CodeQL Analysis** by GitHub: Security analysis from GitHub for C, C++, C#, Go, Java, JavaScript, TypeScript, Python, and Ruby developers. Includes "Configure" and "Code scanning" buttons.
- Veracode Static Analysis** by Veracode: Get fast feedback on flaws with Veracode Static Analysis and the pipeline scanner. Break the build based on flaw severity. Includes "Configure" and "Code scanning" buttons.
- Microsoft C++ Code Analysis** by Microsoft: Code Analysis with the Microsoft C & C++ Compiler for CMake based projects. Includes "Configure" and "Code scanning" buttons.
- Snyk Infrastructure as Code** by Snyk: Detect vulnerabilities in your infrastructure as code files and surface the issues via GitHub code scanning. Includes "Configure" and "Code scanning" buttons.
- Mayhem for API** by ForAllSecure: Automatically test your REST APIs with Mayhem. Includes "Configure" and "Code scanning" buttons.
- mobsf** by mobsf: Mobile Security Framework (MobsF). Includes "Configure" and "Code scanning" buttons.



# Workflow Logs

Build Container  
succeeded 12 days ago in 44s

Search logs

Build Container

- > Set up job 5s
- > Checkout 1s
- > Get Jar file artifact 1s
- > Create container image name 0s
- > GitHub Container Registry Login 1s
- > Setup Docker buildx 7s
- > Cache Container layers 2s

Build and Push Container 21s

```
1 ► Run docker/build-push-action@v2
16 ► Docker info
102 /usr/bin/docker buildx build --build-arg VERSION=1.0.0-fix-error-32853982-
SNAPSHOT --build-arg REPOSITORY_NAME=octodemo/bank-books --build-arg
revision=32853982a076c3ad43b6c9de5428132c62902fd --cache-from
type=local,src=/tmp/.buildx-cache --cache-to type=local,dest=/tmp/.buildx-
cache-new --iidfile /tmp/docker-build-push-vtBsI2/iidfile --tag
ghcr.io/octodemo/bank-books:1.0.0-fix-error-32853982-SNAPSHOT --metadata-
file /tmp/docker-build-push-vtBsI2/metadata-file --push .
103 #1 [internal] load build definition from Dockerfile
104 #1 transferring dockerfile: 1.57kB done
105 #1 DONE 0.0s
106
107 #2 [internal] load .dockerrcignore
108 #2 transferring context: 2B done
109 #2 DONE 0.0s
110
```

github.com

octodemo / bank-books Private generated from octodemo/template-bookstore-v2

Code Issues 1 Pull requests 12 Actions Security 339 Insights Settings

fix sql injection warning Build - Test - Publish #18

Summary

Triggered via push 12 days ago pholleran pushed → 3285398 fix-error Status Success Total duration 2m 22s Billable 4m

Jobs

- Build java 11 on ubuntu-20.04
- Build java 11 on windows-latest
- Build Container

Continuous Delivery Deployment

build\_test\_publish.yml on: push

Matrix: build

```
graph LR; A[Build java 11 on ubuntu-20.04 16s] --> B[Build Container 44s]; C[Build java 11 on windows-latest 1m 11s]
```

Artifacts

Produced during runtime

Name	Size
standalone-ubuntu-20.04-11.jar	13.7 MB
standalone-windows-latest-11.jar	13.7 MB



# Advanced Syntax

## 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

## format

Format outputs

## needs

Make jobs dependent of each other. Share outputs

```
if success() always() cancelled()  
failure()  
Check whether a job should run based on  
variables.
```

## timeout-minutes

Limit runtime

## continue-on-error

Handle termination of workflows

## join

Join arrays into strings

## services

Create sidecar docker images for integration dependencies

## container

Use a container for the steps execution

## contains

Check if a string is contained in another

## startsWith/endsWith

Check start/end of a string

## toJSON/fromJSON

Make string JSON and JSON strings



# Actions Environments and secrets



# Environments

Environments are used to describe a general deployment target like **production**, **staging**, or **development**.



## Dynamic Creation

- Can be dynamically created based on the needs, allowing for flexible and scalable deployment targets.
- Automated setup and teardown of environments, ensuring efficient resource management

## Review & Approval

- Support a review and approval process
- Allows for designated reviewers to approve or reject deployments

## Protection Rules

- Protection rules can be applied to environments, restricting access and enforcing policies
- Can include requirements such as specific branch protections, status checks



# Environments manage and secure deployment targets like production

Control deployments

Add gated deployments with approvals

Control secrets and envs variables

Review all deployments

Navigate directly to urls for deployments

Fully integrated with the checks API

Supports matrix for gated deployments

github.com

## Environments / Configure Production

### Deployment protection rules

Configure reviewers, timers, and custom rules that must pass before deployments to this environment can proceed.

**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...

**Prevent self-review**  
Require a different approver than the user who triggered the workflow run.

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

Enable custom rules with GitHub Apps Beta  
[Learn about existing apps](#) or [create your own protection rules](#) so you can deploy with confidence.

Allow administrators to bypass configured protection rules

**Save protection rules**

### Deployment branches and tags

Limit which branches and tags can deploy to this environment based on rules or naming patterns.

No restriction ▾



# Secrets

GitHub Actions Secrets securely store sensitive data, manage access, and ensure encryption



## Storage & Access

- Allow you to store sensitive information securely at the repo, environment, and org levels
- Access to secrets can be controlled through policies

## Encryption & Security

- Secrets are encrypted before storage & during transmission and storage
- Redacts secrets from logs and allows to use short-lived tokens

## Integration & Management

- Secrets can be integrated into workflows by setting them as inputs or environment variables
- Management of secrets is facilitated through the GitHub REST API and GitHub CLI



# secrets ensure secure handling of credentials and sensitive data across various scenarios

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

The screenshot shows the GitHub 'Secrets' page for a repository. The left sidebar lists various settings like General, Access, Collaborators and teams, Team and member roles, Code and automation, Branches, Tags, Rules, Actions, Webhooks, Copilot, Environments, Pages, Custom properties, Security, Code security and analysis, Deploy keys, and Secrets and variables (which is currently selected). The right side is divided into three main sections: 'Actions secrets and variables', 'Environment secrets', and 'Repository secrets', each listing secrets with columns for Name, Environment, Last updated, and manage buttons.

**Actions secrets and variables**

Secrets and variables allow you to manage reusable configuration data. Secrets are **encrypted** and are used for sensitive data. [Learn more about encrypted secrets](#). Variables are shown as plain text and are used for **non-sensitive** data. [Learn more about variables](#).

Anyone with collaborator access to this repository can use these secrets and variables for actions. They are not passed to workflows that are triggered by a pull request from a fork.

**Environment secrets**

Name	Environment	Last updated
MY_ENV_SECRET	Production	4 minutes ago

**Repository secrets**

Name	Last updated
MY_REPO_SECRET	4 minutes ago

**Organization secrets**

Name	Last updated
MY_ORG_SECRET	4 minutes ago



# Environment secrets

- ✓ Built-in secret store

- ✓ Encrypted (LibSodium sealed box)

# Repository secrets

- ✓ Use directly from your workflow

- ✓ Redacted in workflow logs

# Organization secrets

- ✓ API & CLI support

- ✓ Organization / repository / environment secrets

The screenshot shows the GitHub 'Secrets' page with three main sections: Environment secrets, Repository secrets, and Organization secrets. Each section has a 'Manage [section] secrets' button.

- Environment secrets:** Contains a single secret named 'MY\_ENV\_SECRET' in the Production environment, last updated 4 minutes ago.
- Repository secrets:** Contains a single secret named 'MY\_REPO\_SECRET', last updated 4 minutes ago. It includes edit and delete icons.
- Organization secrets:** Contains a single secret named 'MY\_ORG\_SECRET', last updated 4 minutes ago.

The sidebar on the left lists various GitHub features, with 'Actions and variables' being the active tab. Other tabs include General, Access, Collaborators and teams, Team and member roles, Code and automation, Branches, Tags, Rules, Actions, Webhooks, Copilot, Environments, Pages, Custom properties, Security, Code security and analysis, Deploy keys, and Dependabot.

Section	Secret Name	Environment	Last Updated
Environment secrets	MY_ENV_SECRET	Production	4 minutes ago
Repository secrets	MY_REPO_SECRET		4 minutes ago
Organization secrets	MY_ORG_SECRET		4 minutes ago



# 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
- Enterprise/organization/repository policies for default permissions
- `permissions` syntax for granular permissions on workflow- or job-level
- Can't trigger other workflows



Marketplace Actions exist for integration with other secret stores

```
sample-workflow.yml
name: Pull request labeler
on:
  pull_request:
jobs:
  triage:
    runs-on: ubuntu-latest
    permissions:
      contents: read
      actions: read
      issues: write
    steps:
      - uses: actions/labeler@v2
        with:
          repo-token: ${{ secrets.GITHUB_TOKEN }}
      - uses: myaction@v1
        with:
          mySecret: ${{ secrets.MY_SECRET }}
```



## 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)



# Permissions for GITHUB\_TOKEN

- Token Creation:** Automatically created for each workflow job
- Token Scope:** Limited to the repository containing the workflow
- Modify Permission:** Adjustable in workflow files for specific needs
- Default Permissions:** Admins can set to permissive or restricted
- Additional Permissions:** Use GitHub App or personal access tokens

```
sample-workflow.yml > name
1   name: Pull request labeler
2
3   on:
4     pull_request:
5
6   jobs:
7     triage:
8       runs-on: ubuntu-latest
9
10  permissions:
11    actions: read|write|none
12    checks: read|write|none
13    contents: read|write|none
14    deployments: read|write|none
15    issues: read|write|none
16    packages: read|write|none
17    pull-requests: read|write|none
18    repository-projects: read|write|none
19    security-events: read|write|none
20    statuses: read|write|none
21
22  steps:
23    - uses: actions/labeled@v2
24
25 Workflow permissions
26 Choose the default permissions granted to the GITHUB_TOKEN when running workflows in this organization. You can specify more granular
27 permissions in the workflow using YAML. Learn more about managing permissions.
28 Repository administrators will only be able to change the default permissions to a more restrictive setting.
29
30  Read and write permissions
31   Workflows have read and write permissions in the repository for all scopes.
32  Read repository contents and packages permissions
33   Workflows have read permissions in the repository for the contents and packages scopes only.
34
35 Choose whether GitHub Actions can create pull requests or submit approving pull request reviews.
36  Allow GitHub Actions to create and approve pull requests
37
38 Save
```



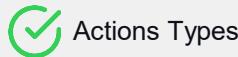
# Building Actions

## Build Actions &

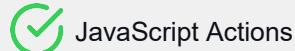
## Workflows



# Write Custom Action



Actions Types



JavaScript Actions



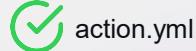
Docker Actions



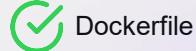
Composite Actions



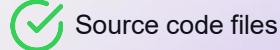
Repository Structure



action.yml



Dockerfile



Source code files

```
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"
```



# JavaScript Action

A JavaScript action is a custom action written in JavaScript or TypeScript that runs directly on the GitHub-hosted runner or a self-hosted runner. It allows you to leverage the full power of Node.js along with GitHub's rich API and automation capabilities. JavaScript actions are ideal for tasks that require high performance and need to interact directly with the GitHub environment.

## action.yml

Metadata File

## index.js

Action Code

## package.json

Package Configuration

The screenshot shows a browser window with the URL [github.com](https://github.com). The page displays two code snippets: a YAML file and a corresponding JavaScript file.

**YAML**

```
name: 'Hello World'
description: 'Greet someone and record the time'
inputs:
  who-to-greet: # id of input
    description: 'Who to greet'
    required: true
    default: 'World'
outputs:
  time: # id of output
    description: 'The time we greeted you'
runs:
  using: 'node20'
  main: 'index.js'
```

**JavaScript**

```
const core = require('@actions/core');
const github = require('@actions/github');

try {
  // `who-to-greet` input defined in action metadata file
  const nameToGreet = core.getInput('who-to-greet');
  console.log(`Hello ${nameToGreet}`);
  const time = (new Date()).toTimeString();
  core.setOutput("time", time);
  // Get the JSON webhook payload for the event that triggered the workflow
  const payload = JSON.stringify(github.context.payload, undefined, 2)
  console.log(`The event payload: ${payload}`);
} catch (error) {
  core.setFailed(error.message);
}
```



# Docker Action

A Docker Action is a custom GitHub Action that runs inside a Docker container. This type of action leverages the capabilities of Docker to create a portable and reproducible environment. Docker Actions are ideal for scenarios where you need a specific software environment, have complex dependencies, or require a high level of isolation.

## Dockerfile

Dockerfile for environments

## action.yml

Action Metadata File

## entrypoint.js

Entrypoint Script

The screenshot shows a GitHub browser window with the URL [github.com](https://github.com). The page displays the configuration for a GitHub Action named 'Hello World'. It includes a YAML file and a Dockerfile.

**YAML**

```
# action.yml
name: 'Hello World'
description: 'Greet someone and record the time'
inputs:
  who-to-greet: # id of input
    description: 'Who to greet'
    required: true
    default: 'World'
outputs:
  time: # id of output
    description: 'The time we greeted you'
runs:
  using: 'docker'
  image: 'Dockerfile'
  args:
    - ${{ inputs.who-to-greet }}
```

**Dockerfile**

```
# Container image that runs your code
FROM alpine:3.10

# Copies your code file from your action repository to the filesystem path `/` of
# the container
COPY entrypoint.sh /entrypoint.sh

# Code file to execute when the docker container starts up (`entrypoint.sh`)
ENTRYPOINT ["/entrypoint.sh"]
```



# Composite Action

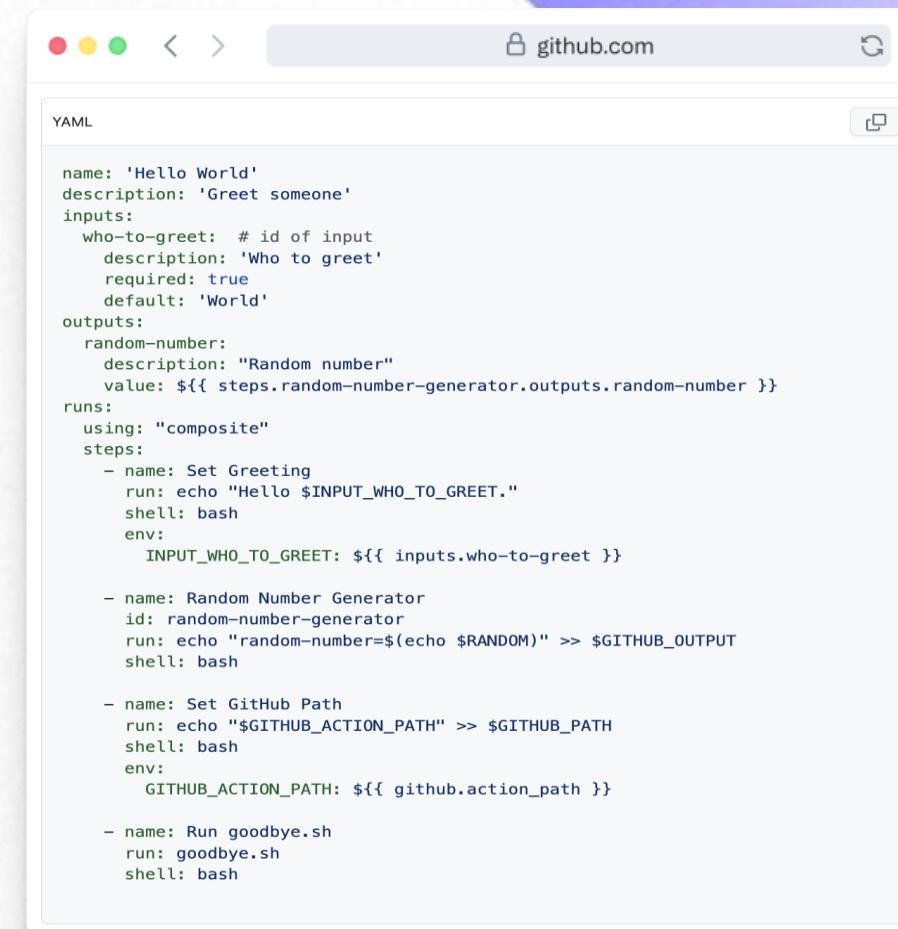
A Composite Action in GitHub is an action that combines multiple individual steps into one reusable component. Instead, they use YAML syntax to define a sequence of steps that are executed together as a single unit. This allows you to encapsulate common tasks and reuse them across different workflows, improving maintainability and reducing redundancy.

## action.yml

## Steps

Metadata File

A series of steps defined in the `action.yml`



The screenshot shows a GitHub browser window with the URL `github.com`. The page displays a YAML configuration file for a composite action named "Hello World". The YAML code defines inputs ("who-to-greet"), outputs ("random-number"), and runs multiple steps: "Set Greeting", "Random Number Generator", "Set GitHub Path", and "Run goodbye.sh". The "Set Greeting" step uses an input from the "who-to-greet" input. The "Random Number Generator" step uses a shell command to generate a random number and output it to \$GITHUB\_OUTPUT. The "Set GitHub Path" step adds the GitHub action path to \$GITHUB\_PATH. The "Run goodbye.sh" step runs a script named "goodbye.sh".

```
YAML

name: 'Hello World'
description: 'Greet someone'
inputs:
  who-to-greet: # id of input
    description: 'Who to greet'
    required: true
    default: 'World'
outputs:
  random-number:
    description: "Random number"
    value: ${{ steps.random-number-generator.outputs.random-number }}
runs:
  using: "composite"
  steps:
    - name: Set Greeting
      run: echo "Hello $INPUT_WHO_TO_GREET."
      shell: bash
      env:
        INPUT_WHO_TO_GREET: ${{ inputs.who-to-greet }}

    - name: Random Number Generator
      id: random-number-generator
      run: echo "random-number=$(echo $RANDOM)" >> $GITHUB_OUTPUT
      shell: bash

    - name: Set GitHub Path
      run: echo "$GITHUB_ACTION_PATH" >> $GITHUB_PATH
      shell: bash
      env:
        GITHUB_ACTION_PATH: ${{ github.action_path }}

    - name: Run goodbye.sh
      run: goodbye.sh
      shell: bash
```



# Best Practices

## Write your own Action



**Creating your own GitHub Actions can significantly enhance the automation of your CI/CD pipelines and improve your workflow efficiency.**

### Security

- Limit Permissions
- Pin Dependencies
- Handle Secrets Securely

### Documentation

- README File
- Provide Examples

### Testing

- Local Testing
- Validation

### Reusability

- Reusable Components
- Composite Actions
- Use GitHub Actions Toolkit

### Optimization

- Efficient Code
- Caching



# Runners

## GitHub Actions Runners



# GitHub Runners

## GitHub Hosted Runners

- **Managed Environment**
  - Maintenance
  - Pre-Installed Software
- **Resource Allocation**
  - Scalability
  - Cost
- **Security**
  - Isolation
  - Network Access
- **Ease of Use**
  - Quick Setup
  - Limited Customization
- **Performance**
  - Standard Performance

## Self-hosted Runners

- **Custom Environment**
  - Full Control
  - Customization
- **Resource Allocation**
  - Scalability
  - Cost
- **Security**
  - Responsibility
  - Network Access
- **Ease of Use**
  - Setup Complexity
  - Flexibility
- **Performance**
  - Custom Performance



# Self-Hosted Runners

[Self-hosted runners](#) are machines that you manage and maintain to execute jobs from GitHub Actions workflows.

## Provision the Runner

## Install the Runner Application

## Configure the Runner

## Manage and Maintain

The screenshot shows the GitHub Self-Hosted Runners configuration interface. At the top, it says "Add new self-hosted runner". It includes sections for "Runner Image" (macOS selected), "Architecture" (x64), and "Download" with a command-line script for provisioning. Below that is a "Configure" section with a command-line script for starting the runner. Further down is a "Using your self-hosted runner" section with a YAML snippet for a workflow file. At the bottom, there's a "Runners" tab with a search bar, a "Standard GitHub-hosted runners" section, and a "ubuntu-latest-m" runner group entry. On the right, there are buttons for "New GitHub-hosted runner" and "New self-hosted runner".

```
# Create a folder
$ mkdir actions-runner && cd actions-runner
# Download the latest runner package
$ curl -o actions-runner-osx-x64-2.317.0.tar.gz -L https://github.com/actions/runner/releases/download/v2.317.0/actions-runner-osx-x64-2.317.0.tar.gz
# Optional: Validate the hash
$ echo "0b23ee79731522d9e1229d14d6c208e06ac9d7ddff5641966209a7708a43c14" actions-runner-osx-x64-2.317.0.tar.gz | shasum -a 256 -c
# Extract the installer
$ tar xzf ./actions-runner-osx-x64-2.317.0.tar.gz
```

```
# Create the runner and start the configuration experience
$ ./config.sh --url https://github.com/enterprises/mousismall --token AW6AFIFVLUYYJ6H275LQUELGMCGE
# Last step, run it!
$ ./run.sh
```

```
# Use this YAML in your workflow file for each job
runs-on: self-hosted
```

Policies Runners Runner groups

Includes all runners across self-hosted and GitHub-hosted runners.

Search runners

New runner

New GitHub-hosted runner  
Pay-as-you-go, customizable, secure, scaled & managed by GitHub

New self-hosted runner  
Bring your own infrastructure

Standard GitHub-hosted runners  
Ready-to-use runners managed by GitHub. [Learn more.](#)

ubuntu-latest-m ubuntu-latest-m  
Runner group: Default Larger Runners Public IP: Disabled



# Security With Self-hosted Runners



**Using self-hosted runners for GitHub Actions offers greater flexibility and control over the environment, but it also introduces additional security considerations.**

## Runner Isolation

- Isolation of Runners
- Ephemeral Runners

## Network Security

- Restricted Network Access
- VPN and VPC

## Access Control

- Least Privilege Principle
- Access Tokens

## Environment

- Secure Configuration
- Monitoring and Logging

## Data

- Secrets Management
- Encryption



# Best Practices With Self-hosted Runners



**Securing self-hosted runners for GitHub Actions requires careful planning and adherence to best practices**

## Runner Lifecycle Management

- Automate Provisioning
- Automate Updates

## Containerization

- Use Containers
- Custom Container Images

## Ephemeral Runners

- Short-Lived Runners
- Scale on Demand

## Audits

- Conduct Audits
- Compliance Checks



# CI/CD

## Action as CI/CD workflows



## Basic CI/CD Action

**GitHub Actions is a powerful feature within GitHub that enables the automation of workflows, particularly for continuous integration (CI) and continuous deployment (CD)**



# Build & Test

This guide shows you how to build, test, and publish a Go package

## Specifying a Go version

## Installing dependencies

## Caching dependencies

## Building and testing your code

```
YAML
name: Upload Go test results

on: [push]

jobs:
  build:

    runs-on: ubuntu-latest
    strategy:
      matrix:
        go-version: [ '1.19', '1.20', '1.21.x' ]

    steps:
      - uses: actions/checkout@v4
      - name: Setup Go
        uses: actions/setup-go@v5
        with:
          go-version: ${{ matrix.go-version }}
      - name: Install dependencies
        run: go get .
      - name: Test with Go
        run: go test -json > TestResults-${{ matrix.go-version }}.json
      - name: Upload Go test results
        uses: actions/upload-artifact@v4
        with:
          name: Go-results-${{ matrix.go-version }}
          path: TestResults-${{ matrix.go-version }}.json
```



# Deploy

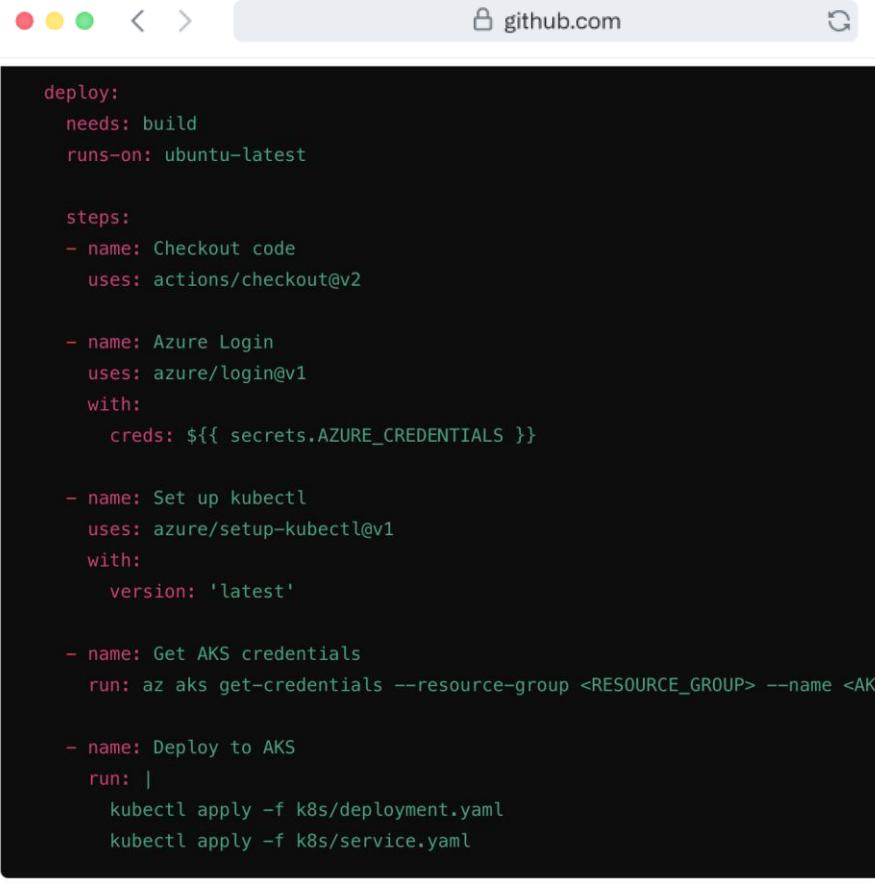
This guide explains how to use GitHub Actions to build and deploy a project to Azure Kubernetes Service

## Azure Login

## Build image on ACR

## Configure deployment

## Deploys application



The screenshot shows a GitHub Actions workflow definition in a browser window. The URL is [github.com](https://github.com). The workflow starts with a deployment step that needs a build and runs on an Ubuntu latest runner. It then performs several steps: checking out code using the actions/checkout@v2 action, performing an Azure login using the azure/login@v1 action with credentials from secrets.AZURE\_CREDENTIALS, setting up kubectl using the azure/setup-kubectl@v1 action with version 'latest', and finally getting AKS credentials using the az aks get-credentials command with resource group and name placeholders. Finally, it deploys to AKS using kubectl apply commands for deployment and service configurations.

```
deploy:
  needs: build
  runs-on: ubuntu-latest

  steps:
    - name: Checkout code
      uses: actions/checkout@v2

    - name: Azure Login
      uses: azure/login@v1
      with:
        creds: ${{ secrets.AZURE_CREDENTIALS }}

    - name: Set up kubectl
      uses: azure/setup-kubectl@v1
      with:
        version: 'latest'

    - name: Get AKS credentials
      run: az aks get-credentials --resource-group <RESOURCE_GROUP> --name <AKS_NAME>

    - name: Deploy to AKS
      run:
        - kubectl apply -f k8s/deployment.yaml
        - kubectl apply -f k8s/service.yaml
```

# GitHub Actions Lab



# GitHub Actions Lab

- Navigate to **node-coverage-lab** repo
- Select “Actions” tab
- Search for node project and select “Node.js” and click configure
- Add step to upload coverage test artifacts(for node version matrix) to repo
- Edit GH Actions file(node.js.yml) directly in browser and commit changes
- Ensure job completed successfully



# Create Node App and Test Coverage

- Create a new GH repo called **node-coverage-lab**
- Launch new Codespace on main branch(green button)
- When Codespace loads, run the following commands:
- `npm init -y`
- `npm install --save-dev jest`
- Verify node version: `node --version`
- Create new file **app.js**(add supplied code)
- Create a new folder at the root called **test**
- Create a new file **app.test.js**(add supplied code)
- Run `git add .`, `git commit -m “ ”`, `git push origin main`
- Run `npm test`



# GitHub Advanced Security



# Challenges and Opportunities

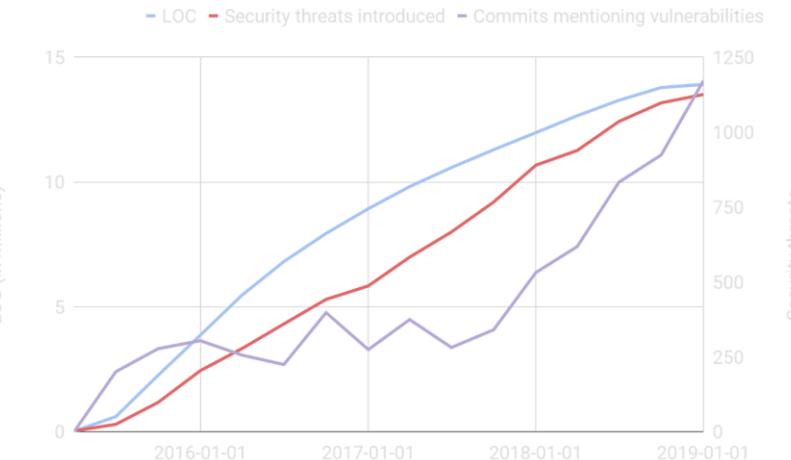
---

Industry challenges and insights



**Despite increasing developer awareness, security threats continue to rise**

Security threats continue to rise with LOC



# State of AppSec

Despite billions of dollars of investment...

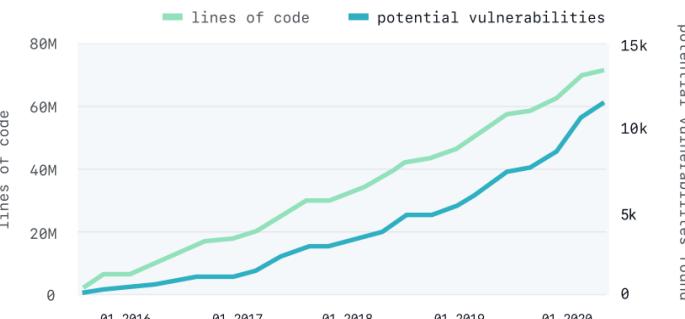


85% of applications still contain a security issue



Code written in 2020 is just as likely to introduce a security issue as code written in 2016

Potential vulnerabilities found in source code scale with lines of code written



**Flaws in  
applications are  
consistently the  
#1 attack vector  
for breaches**

# **State of AppSec**

**is falling further behind the current state of Development**



1:100 Security team members to developers



Lack of knowledge voted the main AppSec challenge



Remediation trends are stagnant

# Everyone wants to shift security left...



# Solution: Shift security left, but how?

**800x** more developers than security researchers



Professional Developers

~ 56M

~ 70K

Security Researchers

# The GitHub Difference

Developer empowered security to fix in minutes not months

## Native

- Embedded from init to ship
- Automatically deployed
- Detailed changelog
- Visibility into security posture across all code

## Fix in minutes

- In-context feedback
- Automated fixes
- Jargon free language
- Blameless culture



## Trust-By-Design

- Static code analysis
- Supply chain analysis
- Secret scanning
- Security Lab
- Extensible platform

## Community Driven

- Supported by millions of developers and security researchers
- Contributions from leading peers



# Native Application Security

“

Security at the expense of usability comes at the expense of security.

Lee Cookson  
Engineering Director |  
@ Dow Jones

Secure code without disrupting innovation with security testing embedded in the developer experience



**Embedded from init to ship**



**Automatically deployed** to immediately increase code coverage



**Detailed changelog** demystifying progress and simplifying audit/collaboration



**Security overview** provides visibility into security posture across all code

# Trust-by-design

“

GitHub helps us ensure that we have our security controls baked into our pipelines all the way from the first line of code you're writing.

**Miguel El Lakkis**

Chief Information Security Officer |  
@ Dow Jones

Trust your software with an end-to-end security solution for securing your software supply chain and code, that developers actually use and love!



Static code and supply chain analysis



Secret detection and leak prevention



Extensible platform with SARIF support

# Fix in Minutes

“

Our engineering teams place tremendous value in GitHub's latest security features like vulnerability alerts and automated fixes.

Jon Parise

Engineering Architect |

@ Pinterest

Empower developers with more accurate results and fix the vulnerabilities that matter the most for your organization



**In-context feedback and automated fixes** where possible leading to industry leading fix rates and MTTR



**Jargon free language** that empowers developers to fix without security intervention



**Blameless developer trusted culture** that leads to higher adoption of fixes

# Community Driven

Benefit from thousands of analyses created by GitHub's team of security researchers and language experts and contributions from other GitHub users—such as leading security researchers at Microsoft, Google, Uber, and others.

Scale your security team with the World's Leading Security Researchers



Scale your security team with a **dedicated security team that also leverages the world's expert community of developers**



Open source foundations **encourage collaboration and community engagement**



**Benefit from user driven design and frequent innovation**

# Secure by design

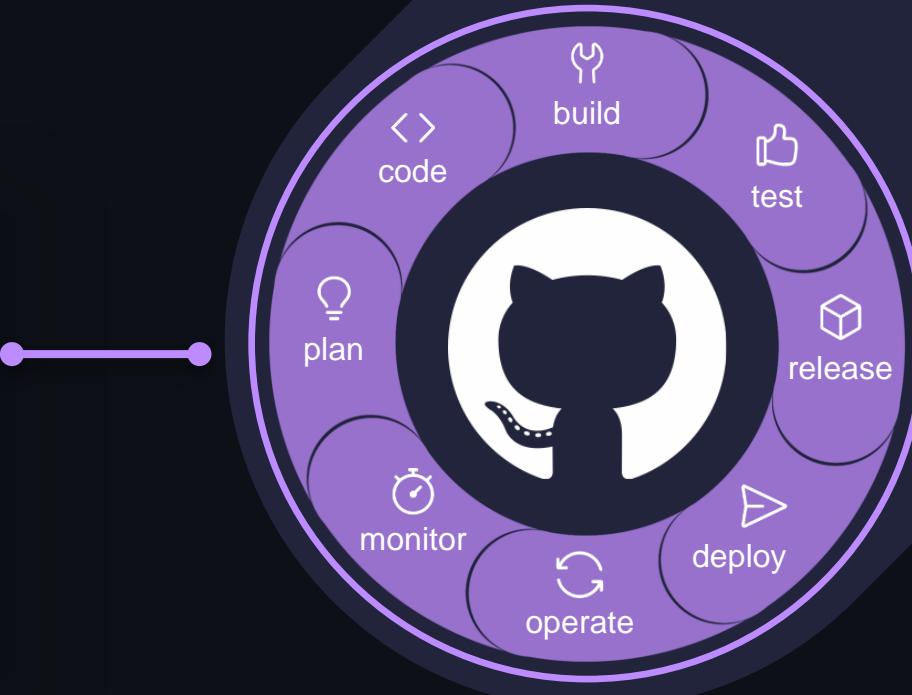
---

Secure development end to end



# Security built into the Developer Lifecycle

A screenshot of a GitHub pull request page. The title is "Allow the webhook to send a base64 gzip sarif file #33". The commit message is "arthurnn wants to merge 2 commits into `master` from `arthurnn/zip_base64`". The pull request has 4 checks and 2 files changed. The code review shows a diff of a file named `app/controllers/webhooks_controller.rb`. A specific line of code is highlighted with a red bar, indicating a security issue: "17 - ~~sarif\_json = JSON.parse(check\_run['text'])~~". A tooltip for this line reads: "Improper Neutralization of Directives in Dynamically Evaluated Code. The software receives input from an upstream component, but it does not neutralize or incorrectly neutralizes code syntax before using the input in a dynamic evaluation call". The pull request has 0 reviews and 0 comments.



# Community-powered security and compliance



## Dependency insights

- Real-time inventory
- License compliance
- Vulnerability alerting



## Vulnerability Management

- Automated code scanning
- Private secret scanning
- Largest vulnerability database
- Automated security updates



## CodeQL

- World's most advanced code analysis
- Vulnerability hunting tool
- Community of top security experts



# GitHub Advanced Security Impact

Code vulnerability  
fix rate of

72%

compared to

15%

Industry norm after 7 days



OSS vulnerabilities  
MTTR decreased from  
180 days to 40 days  
with Dependabot

GHAS scan and alerts  
for secrets and tokens  
from 60+ providers  
and growing



# Dependabot

## Automated pull requests for security & version updates



Keep your projects secure and up to date by monitoring them for vulnerable and out-of-date components. If a suggested update is found, we'll automatically open a pull request with suggested fixes.

## Integrated with developer workflow

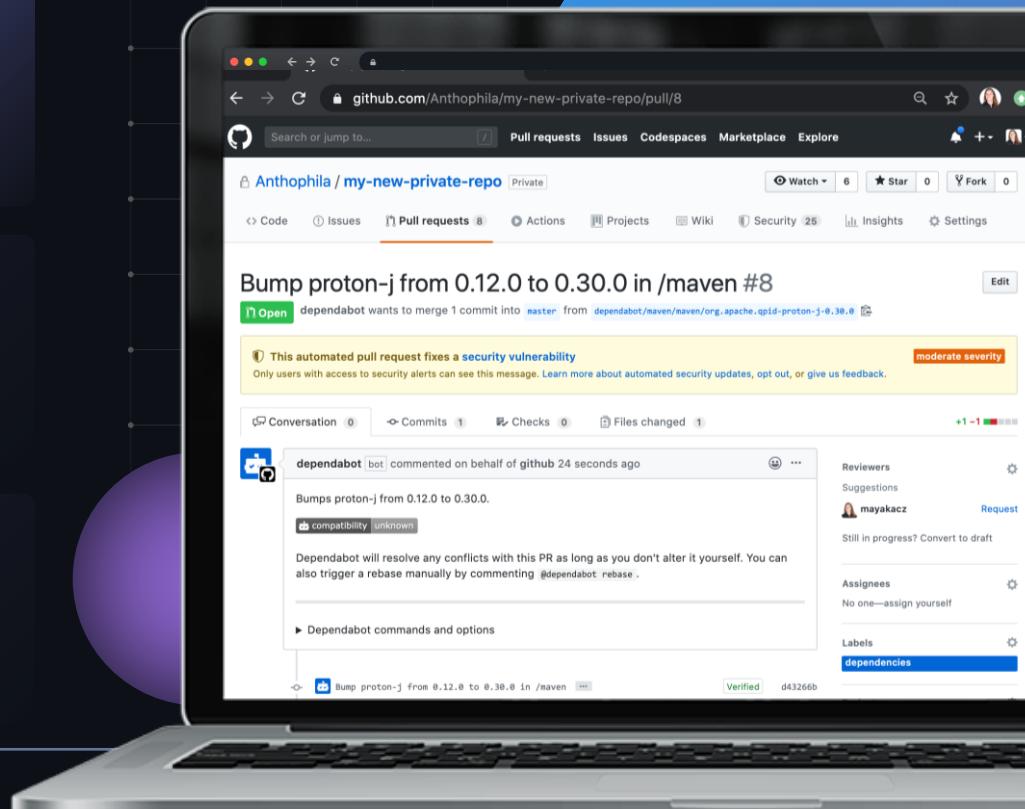


Dependabot is integrated directly into the developer workflow for a frictionless experience and faster fixes.

## Rich vulnerability data



GitHub tracks vulnerabilities in packages from supported package managers using data from security researchers, maintainers, and the National Vulnerability Database—all discoverable in the GitHub Advisory Database.



# Code scanning



## Find and fix vulnerabilities fast

Find and fix vulnerabilities before they are merged into the code base with automated CodeQL scans.



## Community of top security experts

Your projects are powered by world-class security teams. Use queries created by the security community in your projects.



## Integrated with developer workflow

Integrate security results directly into the developer workflow for a frictionless experience and faster development

The screenshot shows a GitHub repository named 'dsp-testing / code-scanning-demo'. The 'Security' tab is selected. On the left, there's a sidebar with sections like Overview, Security policy, Security advisories, Dependabot alerts, Code scanning alerts (which has one item highlighted), and Detected secrets. The main area displays a code editor with a file named 'test.ts'. A specific line of code is highlighted in yellow: 'res.setHeader('Location', url);'. Below the code, a tooltip provides context: 'Untrusted URL redirection due to user-provided value. CodeQL'. The tooltip also includes the tool name 'CodeQL', rule ID 'js/server-side-unvalidated-url-redirection', and a 'Query View source' button. At the bottom of the tooltip, a note reads: 'Directly incorporating user input into a URL redirect request without validating the input can facilitate phishing attacks. In attacks, unsuspecting users can be redirected to a malicious site that looks very similar to the real site they intend to visit which is controlled by the attacker.' There are 'Show more' and 'View source' buttons at the bottom right of the tooltip.

# Secret scanning



## Identifies secrets as early as possible

Finds secrets (including Azure secrets) the moment they are pushed to GitHub and immediately notifies developers when they are found.



## Community of secret scanning partners

For every commit made to your repository and its full git history, we'll look for secret formats from secret scanning partners.



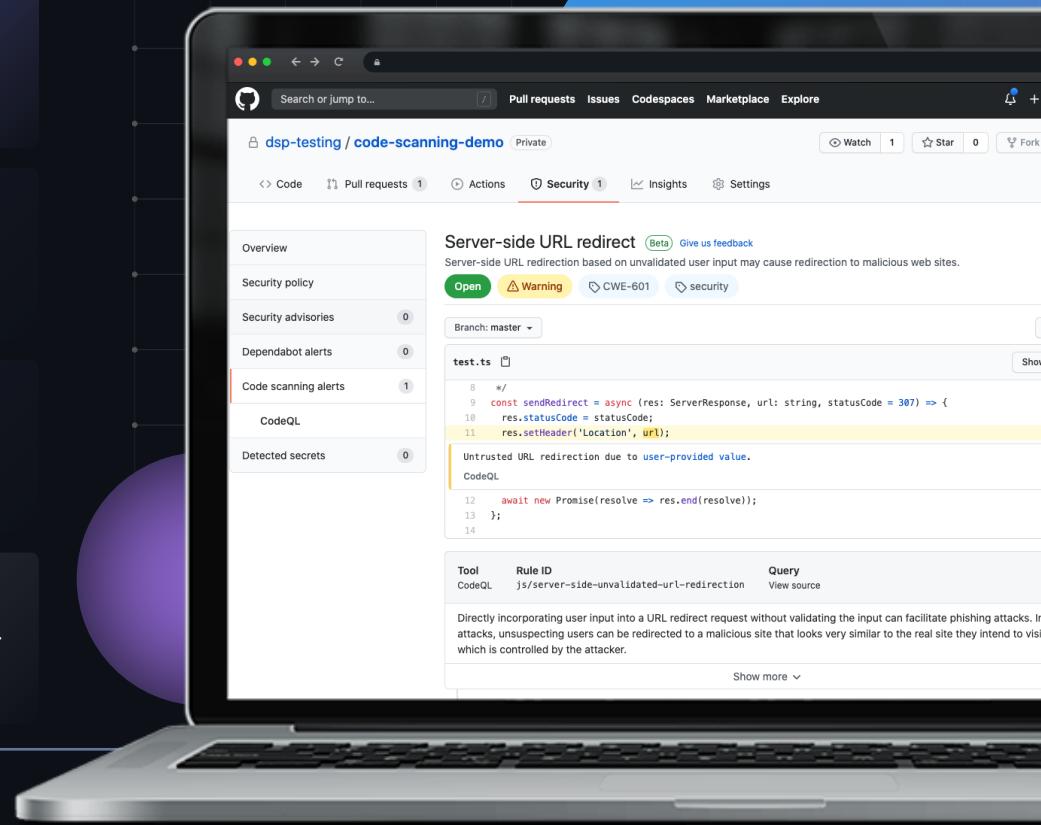
## Define custom patterns

Scan for patterns that are internal to your organization across your repositories.



## Supports both public and private repos

Secret scanning watches both public and private repos for potential secret vulnerabilities.



# Security Overview



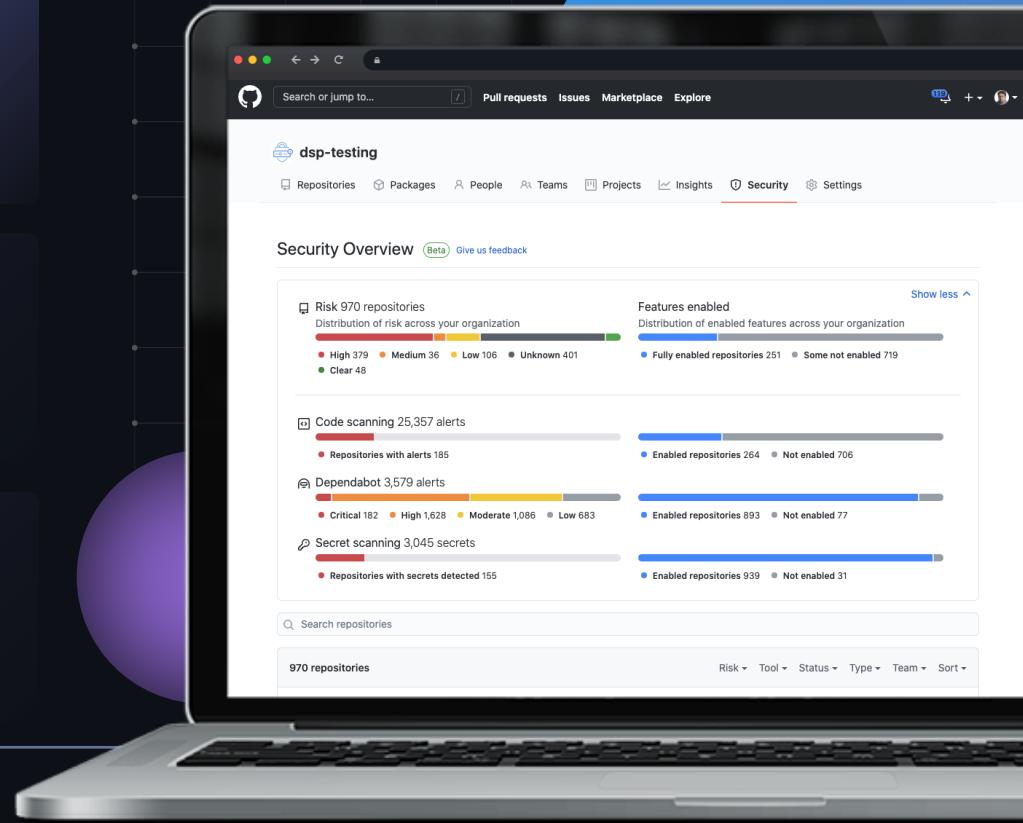
A single place to view code scanning, secret scanning and Dependabot alerts across your codebases



Determine the riskiest repos with high numbers of unaddressed alerts



Monitor enrollment status as you rollout security features across your organization



# GitHub Advanced Security Lab



# GitHub Advanced Security - Juice-Shop

- Navigate to <https://github.com/juice-shop/juice-shop>
- Fork the repo with your personal GitHub Account
- Click on the **Security** tab
- Click on **Dependabot** and enable(along with Dependency graph)
- Scroll down to **Code Scanning -> CodeQL Analysis -> Default**
- After waiting a few minutes, navigate back to Security and view Dependabot and Code scanning results.



# GitHub Copilot



# GitHub Copilot Fundamentals

What is GitHub Copilot?

What is required?

Supported IDEs

Data flow

“



GitHub Copilot



GitHub Copilot is an AI-powered code completion tool. It's designed to assist developers by providing suggestions for whole lines or blocks of code as they type. It works directly in your editor, helping you to test out new ideas faster, write code more efficiently, and learn new APIs or languages on the go.

- *Copilot Chat*



# What is Required ?



GitHub account



Copilot license



Supported IDE



# Supported IDEs



Visual Studio



VS Code



JetBrains IDE Suite



Vim/Neovim



Azure Data Studio



Xcode



# How Copilot Fulfills a Request

Apply several **pre-model filters**

- Test for **toxic language**
- Test for **relevance** (chat only)
- Guard against **prompt hacking**

Copilot assembles context from tabs open in the code editor



OpenAI  
model

Context / Prompt  
N Suggestions

Proxy

Context / Prompt  
N Suggestions



Copilot  
Agent

Context / Prompt  
N Suggestions

Apply several **post-model filters**

- **Code quality**
- **Unique Identifiers** (eg.: emails, IPs etc.)
- Suggestion **matching public code**

A screenshot of a code editor interface showing a file named 'calculator.js'. The code defines a 'calculator' module with methods for addition, subtraction, multiplication, division, and exponentiation. It includes parameter annotations like `param {number} num1` and `param {string} operator`. A red box highlights the first few lines of the code, and a dashed blue line indicates the continuation of the file.

```
1 /**
2  * @description: A calculator module that can add, subtract, multiply, divide and exponent by taking in two numbers
3  * @param {number} num1
4  * @param {number} num2
5  * @param {string} operator
6  *
7  * @returns {number} result of the operation
8  */
9
10 // the calculator function
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
```

Copilot



# Feature overview

- GitHub Copilot for Business - Features
- GitHub Copilot for Enterprise - Features
- GitHub Copilot - New Features
- GitHub Copilot Roadmap

# IDE Features

						
Code completion	✓	✓	✓	✓	✓	✓
Copilot Chat	✓	✓	✓			
Extensions	✓	✓	✓			
Knowledgebase	✓	✓				
Enhanced Skills	✓					
Code Review	✓					
Feature-rich	✓					
Custom Instructions	✓					



# GitHub Copilot Business- Feature Overview

GA

GA

GA

## Code completion

Code completion and suggestions

## Copilot Chat

Receive answers to coding-related questions

## Copilot CLI

Translate natural language into shell commands

# GitHub Copilot Business- Feature Overview

GA

GA

GA

## PR Summaries

Create summary of changes intro

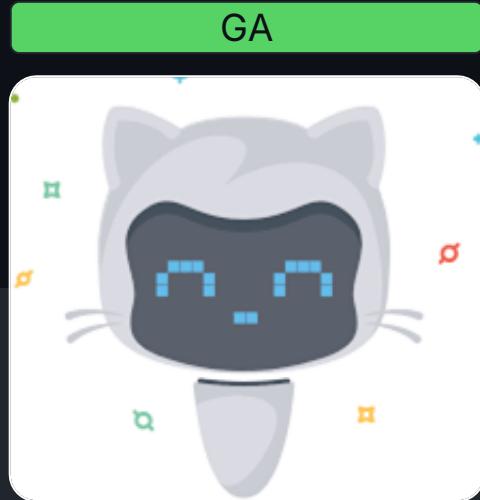
## Chat in the GUI

Search documentation directly from the UI

## Content Exclusion

Exclude sensitive data to ensure secure AI code generation.

# GitHub Copilot Business- Feature Overview



GitHub Skills

Extend the functionality of Copilot Chat with Skills.



Model Picker

Bringing developer choice to Copilot with new models.



Private Extensions

Integrate your custom dev tools directly into Copilot Chat!

# GitHub Copilot Enterprise- Feature Overview

Enterprise - GA

Enterprise - GA

Enterprise - GA



## Knowledge bases

Leverage Organizational  
Context in Chat

## Issues & Discussions

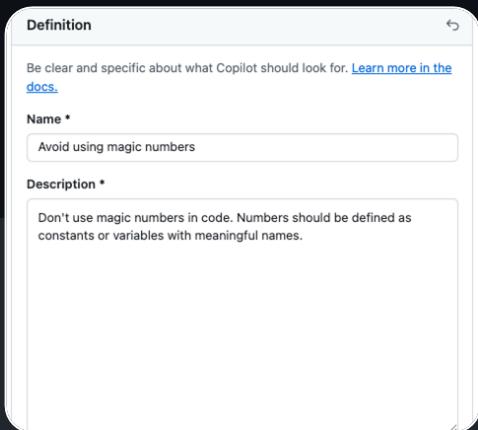
Understand context  
from your Issues &  
Discussions with Chat!

## Enterprise Settings

Access to Enterprise  
Level Content  
Exclusion, Settings &  
Enterprise API Metrics

# GitHub Copilot Enterprise- Feature Overview

## Enterprise - GA



### Coding Guidelines

Enhance Code consistency and quality with coding standards

## Enterprise - Private Preview



### Copilot Workspaces in PRs

Accelerate code reviews and improve collaboration within pull requests

## Enterprise - Public Preview



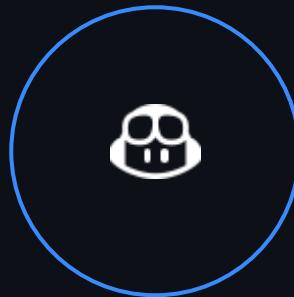
### Fine-Tuning

Train Copilot on your own data

# Recently Added Features



Pull Request (PR)  
Summaries



Copilot Chat in  
GitHub.com



# Generating a pull request summary

Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#). Learn more about diff comparisons [here](#).

base: main ▾ ← compare: CallMeGreg/2FA-support ✓ Able to merge. These branches can be automatically merged.

Add a title

Add 2FA Support

Add a description

Write Preview

Add your description here...

Markdown is supported Paste, drop, or click to add files

Create pull request

① Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

Reviewers

No reviews

Assignees

No one—assign yourself

Labels

None yet

Projects

None yet

Milestone

No milestone

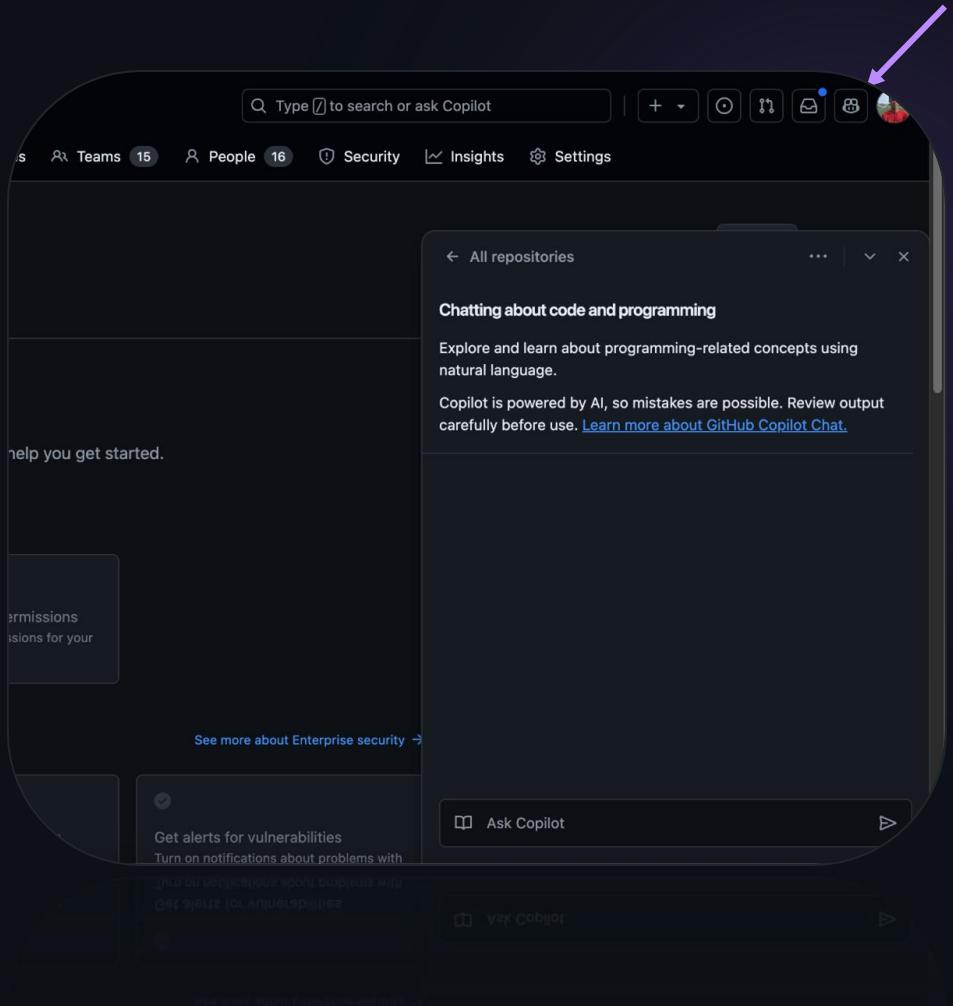
Development

Use [Closing keywords](#) in the description to automatically close issues

Helpful resources

[GitHub Community Guidelines](#)





# Copilot Chat in GitHub.com

- Index repositories to improve your suggestions
- Reference code, pull requests, issues (and more!) as context
- Search the web with less context switching

# Indexing repositories

The screenshot shows a GitHub repository page for 'app-abc-repo' in the 'callmegreg-demo-org' organization. The repository has 14 branches and 0 tags. The commit history lists 26 commits from 'CallMeGreg' over the last year, including initial commits for various services like '.devcontainer', '.github', 'authn-service', 'exercises', 'frontend', 'gallery-service', 'scripts', 'storage-service', and updates to 'LICENSE.md' and 'README.md'. The repository has no description, website, or topics provided. It includes sections for Readme, MIT license, Activity, Custom properties, 0 stars, 0 watching, and 0 forks. There are also sections for Releases (no releases published) and Packages (no packages published).

callmegreg-demo-org / app-abc-repo

Type  to search or ask Copilot

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

app-abc-repo Internal generated from [ghas-bootcamp/ghas-bootcamp](#)

Edit Pins Watch Fork Star

master 14 Branches 0 Tags Go to file Add file Code

CallMeGreg Merge pull request #15 from callmegreg-demo-org/CallMeGre... 871ca82 · 4 months ago 26 Commits

File	Message	Time
.devcontainer	Initial commit	last year
.github	Update central-scan.yml	5 months ago
authn-service	Update requirements.txt	last year
exercises	Initial commit	last year
frontend	Initial commit	last year
gallery-service	Initial commit	last year
scripts	Create metadata_helpers.py	last year
storage-service	Initial commit	last year
LICENSE.md	Initial commit	last year
README.md	Update README.md	last year
example.kt	Create example.kt	last year

About

No description, website, or topics provided.

Readme

MIT license

Activity

Custom properties

0 stars

0 watching

0 forks

Releases

No releases published

Create a new release

Packages

No packages published

Publish your first package

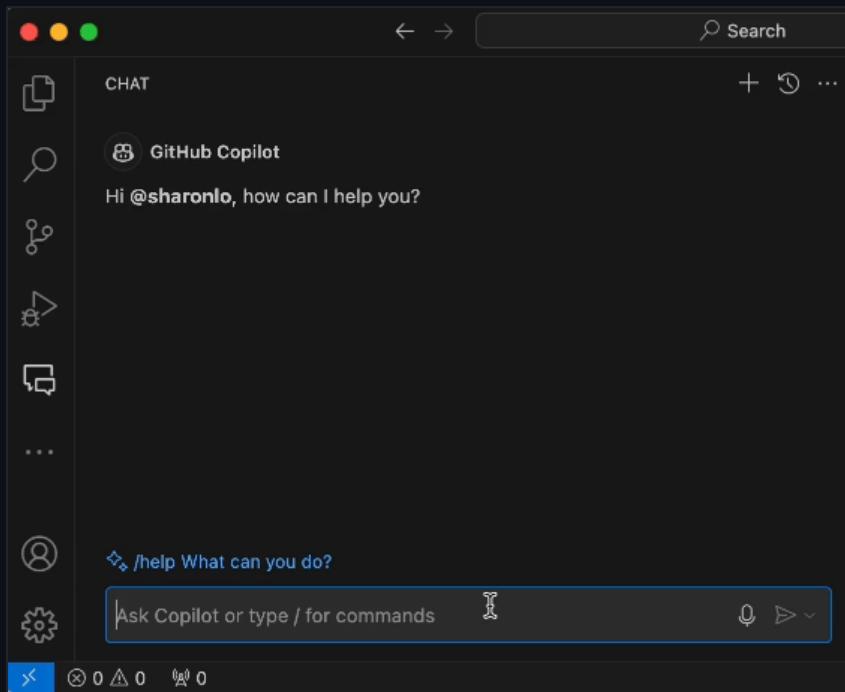


# Copilot Chat is powered by “skills”

The screenshot shows the GitHub Copilot Chat interface. At the top, there's a header with a back arrow labeled "All repositories" and a three-dot menu icon. Below the header, a repository card for "Chatting about octodemo/beat-bot" (Internal repository) is displayed. A note states: "Copilot is powered by AI, so mistakes are possible. Review output carefully before use. [Learn more about GitHub Copilot Chat.](#)" A green bullet point indicates it's "Indexed for improved understanding and accuracy." Below this, a section titled "Ask about the repository:" lists three questions with right-pointing arrows: "What questions can I ask?", "Can you tell me about this repository?", and "How should I get started exploring this repo?". At the bottom, there's a large input field with the placeholder "Ask Copilot" and a blue send button with a right-pointing arrow. Below the input field are icons for a file, a user profile, and an '@' symbol, along with the text "↑/↓ to cycle previous inputs".



# Search the web in real time with Bing



# GitHub Copilot - What's coming up?

The screenshot shows a dark-themed web browser window displaying the GitHub Next projects page at <https://githubnext.com/#projects>. The page is titled "Projects". It features four project cards:

- Code Atlas** (June 28, 2023, NAPKIN SKETCH): How can we make LLM responses more robust and easier to understand by combining their fluid reasoning with rigid structure?
- GPT-4 with Calc** (May 11, 2023, COMPLETED): An exploration of using calculation generation to improve GPT-4's capabilities for numeric reasoning.
- Copilot for Docs** (March 22, 2023, WAITLIST): How would it feel to have an expert on hand at all times? We built a tool that uses relevant information
- Copilot for Pull Requests** (March 22, 2023, WAITLIST): Pull requests are a central part of the GitHub user experience. Copilot for PRs brings the power of



\*Note\* - This page is constantly updated. View <https://githubnext.com/#projects> for up to date info.

# Coding



# Copilot vs Copilot Chat

## Copilot

Direct Code Writing

---

Seamless IDE Integration

---

Solo Development

## Copilot Chat

In-Depth Assistance

---

Learning & Teaching

---

Collaborative Scenarios



# Large Language Models (LLMs) Limitations & caveats



GitHub Copilot leverages LLMs in order to produce output. Some general caveats related to LLMs should be observed when using GitHub Copilot or other LLM-based tools.

**Outdated or Incomplete Training Data:** LLMs are trained on historical data and may not account for newer language versions, frameworks, or libraries, leading to outdated or inaccurate suggestions (note: GitHub Copilot does have features to remedy this risk such as support for real-time online search).

**Quality of Suggestions:** Libraries, frameworks, and languages are not equally represented in the training data which may impact the quality of the suggestions.

**Hallucinations:** LLMs can produce output that are “hallucinations” i.e. responses presented as accurate but that are not. Always inspect the output! (more on how to remedy this later)



# Using Copilot

## GitHub Copilot

- Code Driven Development
- Multilingual
- Offering Alternative Results
- Create Unit Tests
- Documentation



# Good code techniques

# Guide Copilot

Use good names

GitHub Copilot understands  
natural language

Spell out variable names

Single letter variables and  
abbreviations are ambiguous

Keep functions functional

Follow strong principles  
when creating named blocks

Be consistent

Generated code follows  
contextual patterns



# Helpful Patterns

## Variable names

Use descriptive variable names to make your intentions clear.

```
total_attendees = 5
```

## Method Signatures

Define method signatures with unambiguous parameter names and types.

```
calculateAverage(int num)
```

## Naming Conventions

Maintain consistent naming conventions for variables and functions

i.e. using **camelCase** for variable names consistently



# Helpful Patterns (Cont.)

## Input/Output Format

Describe the expected input and output formats.

"Write a function that takes an array of strings as input and returns true if a palindrome is found."

## Error Handling

Specify error handling scenarios

"Exit where the integer input is empty and throw an error if the input is not an integer at all"

## Control Structures

Describe control flow structures.

"Write a while loop to find the first prime number given a list of integers. If no such number exists, the loop should exit."



# Advanced Settings

## Configuration Options



-  Status: Ready
-  GitHub Copilot Chat
- Open Completions Panel...
- Disable Completions
- Disable Completions for 'markdown'
-  Edit Keyboard Shortcuts...
-  Edit Settings...
- Show Diagnostics...
- Open Logs...

[Configuring GitHub Copilot in your environment](#)



# Using Copilot Chat

## GitHub Copilot Chat

- Refactoring Code
- Generating Tests
- Debugging
- Creating a Workspace
- Documentation



# Copilot Chat: Slash Commands

/help to find  
available  
commands in your  
IDE



# In-file Copilot Chat

Copilot offers  
**in-file Copilot  
feature** to  
selectively  
improve

GitHub Copilot Chat (default)



# Run in Terminal

Ask in GitHub  
Copilot CLI

GitHub Copilot Chat (default)

```
→ copilot gh copilot explain "upgrade gh cli on macos"]
```



# Copilot Prompt Engineering

## What

Prompt engineering is the process of designing and creating high-quality prompts that can be used to generate accurate and useful code suggestions with Copilot.

## Why

- Maximizes utility of Copilot
- Minimizes repeated iterations
- Determines code suggestion quality
- Skill prompts guide Copilot to understand context & nuances
- Refined interactions reduces misunderstandings

## How

- Neighboring Tabs
- Zero-Shot Prompting
- One-Shot Prompting
- Few-Shot Prompting
- Let's Think Step by Step



# Prompting Best Practices

## Improving results



These are best practices, now let's see how those can be used in different ways



### Provide references

Improve relevance of the response by providing an example and context



### Write clear instructions

Refine your prompt, provide context, write clearly, and give Copilot ample input for better results



### Split up big tasks

Breaking down complex tasks minimizes errors and utilizes previous outcomes for efficiency



### Allow time to think

Requesting Copilot's thought process will enhance Copilot accuracy, but it may prolong wait times.



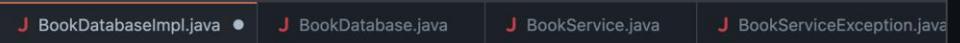
### Test changes systematically

Measure performance, watch for prompt changes' side effects, and use test suites for implementation



# Neighboring Tabs

Keep relevant files open, things related to your current code



```
BookDatabaseImpl.java • BookDatabase.java BookService.java BookServiceException.java
src > main > java > com > github > demo > service > BookDatabaseImpl.java > BookDatabaseImpl > getBooksByAuthor
145
146     @Override
147     public List<Book> getBooksByAuthor(String author) throws BookServiceException {
148         // 1.create list of type Books
149         // 2.create Prepared statement and query to get books by Author
150         // 3.iterate through results and add to list
151         // 4.close connection if prepared statement is not null
152         // 5.return list
153
154         List<Book> books = new ArrayList<Book>();
155
156         if (!isValid()) {
157             throw new BookServiceException(message:"Database connection is not valid, check");
158         }
159
160         PreparedStatement ps = null;
161
162         try {
163             ps = connection.prepareStatement(sql:"SELECT * FROM books WHERE author LIKE ?");
164             ps.setString(parameterIndex:1, "%" + author + "%");
165
166             ResultSet results = ps.executeQuery();
167
168             while (results.next()) {
169                 Book book = new Book(
170                     results.getString(columnLabel:"author"),
171                     results.getString(columnLabel:"title"),
172                     results.getString(columnLabel:"image"));
173                 books.add(book);
174             }
175         } catch (SQLException e) {
176             throw new BookServiceException(message:"Error occurred while executing query");
177         } finally {
178             if (ps != null) {
179                 ps.close();
180             }
181         }
182     }
183 }
```



If at first you  
don't succeed,  
iterate!

## Zero Shot Prompt

```
1 //Write a JavaScript function that finds
2 //the maximum value in an array.
3 function max(array) {
4     return Math.max.apply(null, array);
5 }
```

## One-Shot Prompt

```
1 //Create a JavaScript function, `findMax`, that takes an array of
2 //and returns the string with the maximum length.
3 //Ensure that the function works correctly for arrays that are em-
4 //as well as arrays that contain both strings and numbers.
5 function findMax(array) {
6     var max = array[0].length;
7     array.map(v => max = Math.max(max, v.length));
8     result = array.filter(v => v.length == max);
9     return result;
10 }
```



# Provide a high level description to clarify expectations

```
calculations.py > ...
1  """
2  Create a Python function, `sum_array_values`, that finds
3  and returns the sum of integers in an array.
4
5  1.Verify if the array is empty and exit if it is.
6  2.Throw error if the array contains non-integer values.
7  3.Return the sum of the array values if the array is valid.
8
9  Please consider the following example data:
10 Example 1: [5, 10, 15, 20, 8]
11 Example 2: [2, 4, 6, 1, 9, 12]
12 The generated code should work correctly for arrays like above
13 """
14 #define sum_array_values function
15 def sum_array_values(array):
16     # 1.Verify if the array is empty and exit if it is.
17     if len(array) == 0:
18         return 0
19     # 2.Throw error if the array contains non-integer values.
20     for i in array:
21         if type(i) != int:
22             raise TypeError("The array contains non-integer values.")
23     # 3.Return the sum of the array values if the array is valid.
24     return sum(array)
```



# Secure coding



# Copilot and Secure Coding

- AI-based vulnerability system that helps prevent insecure coding patterns (e.g. SQL script injection)
- Vulnerability filters are applied only to the Copilot-generated suggestion itself
  - ✖ It cannot detect downstream vulnerabilities introduced by the code e.g. on deployment infrastructure
  - ⓘ We recommend taking the **same precautions** you take with code written by engineers (linting, code scanning, etc.)
- Copilot Chat can be used to query code for known vulnerabilities



# Security & Trust

## Copilot Trust Center

- Security
- Privacy
- Data flow
- Copyright
- Labor market
- Accessibility
- Contracting

The screenshot shows the GitHub Copilot Trust Center homepage. At the top, there's a navigation bar with links for Resources, Why GitHub, Collections, Topics, Type, a search icon, Enterprise trial, and Contact Sales. The main title "GitHub Copilot Trust Center" is prominently displayed. Below the title, a subtitle reads: "We enable developers and organizations to maximize their potential by prioritizing security, privacy, compliance, and transparency as we develop and iterate on GitHub Copilot." There are five categories listed below: Security, Privacy, IP and Open Source, Labor Market, and Accessibility. On the right side, there's a video player titled "Trust Center Overview" showing a video thumbnail with a white cat silhouette and a red play button. A black arrow points to the "Watch on YouTube" button at the bottom of the video player.



# GitHub Copilot Lab – Copilot Adventure

- <https://microsoft.github.io/CopilotAdventures/>
- Challenges can be done in C#, JS, or Python
- Start with the Warmup Adventure
- Select one of the beginner adventures
- Use either GH Codespaces or locally with VS Code





# Thank you

