**Github Workflow**

**workflows**

Get a high-level overview of GitHub Actions workflows, including triggers, syntax, and advanced features.

[**About workflows**](https://docs.github.com/en/actions/writing-workflows/about-workflows#about-workflows)

A **workflow** is a configurable automated process that will run one or more jobs. Workflows are defined by a YAML file checked in to your repository and will run when triggered by an event in your repository, or they can be triggered manually, or at a defined schedule.

Workflows are defined in the .github/workflows directory in a repository. A repository can have multiple workflows, each which can perform a different set of tasks such as:

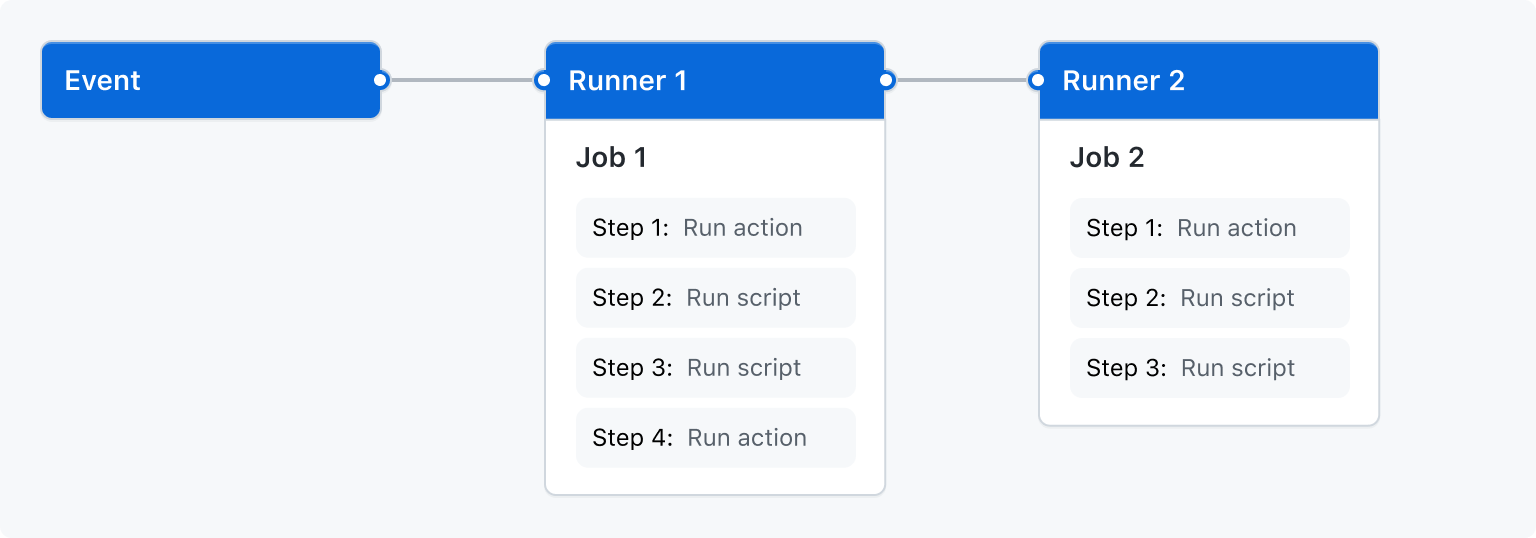
* Building and testing pull requests.
* Deploying your application every time a release is created.
* Adding a label whenever a new issue is opened.

[**Workflow basics**](https://docs.github.com/en/actions/writing-workflows/about-workflows#workflow-basics)

A workflow must contain the following basic components:

1. One or more *events* that will trigger the workflow.
2. One or more *jobs*, each of which will execute on a *runner* machine and run a series of one or more *steps*.
3. Each step can either run a script that you define or run an action, which is a reusable extension that can simplify your workflow.

For more information on these basic components, see "[Understanding GitHub Actions](https://docs.github.com/en/actions/learn-github-actions/understanding-github-actions#the-components-of-github-actions)."



[**Triggering a workflow**](https://docs.github.com/en/actions/writing-workflows/about-workflows#triggering-a-workflow)

Workflow triggers are events that cause a workflow to run. These events can be:

* Events that occur in your workflow's repository
* Events that occur outside of GitHub and trigger a repository\_dispatch event on GitHub
* Scheduled times
* Manual

For example, you can configure your workflow to run when a push is made to the default branch of your repository, when a release is created, or when an issue is opened.

For more information, see "[Triggering a workflow](https://docs.github.com/en/actions/using-workflows/triggering-a-workflow)", and for a full list of events, see "[Events that trigger workflows](https://docs.github.com/en/actions/using-workflows/events-that-trigger-workflows)."

[**Using workflow templates**](https://docs.github.com/en/actions/writing-workflows/about-workflows#using-workflow-templates)

GitHub provides preconfigured workflow templates that you can use as-is or customize to create your own workflow. GitHub analyzes your code and shows you workflow templates that might be useful for your repository. For example, if your repository contains Node.js code, you'll see suggestions for Node.js projects.

These workflow templates are designed to help you get up and running quickly, offering a range of configurations such as:

* CI: [Continuous Integration workflows](https://github.com/actions/starter-workflows/tree/main/ci)
* Deployments: [Deployment workflows](https://github.com/actions/starter-workflows/tree/main/deployments)
* Automation: [Automating workflows](https://github.com/actions/starter-workflows/tree/main/automation)
* Code Scanning: [Code Scanning workflows](https://github.com/actions/starter-workflows/tree/main/code-scanning)
* Pages: [Pages workflows](https://github.com/actions/starter-workflows/tree/main/pages)

Use these workflows as a starting place to build your custom workflow or use them as-is. You can browse the full list of workflow templates in the [actions/starter-workflows](https://github.com/actions/starter-workflows) repository. For more information, see "[Using workflow templates](https://docs.github.com/en/actions/writing-workflows/using-starter-workflows)."

[**Advanced workflow features**](https://docs.github.com/en/actions/writing-workflows/about-workflows#advanced-workflow-features)

**This section briefly describes some of the advanced features of GitHub Actions that help you create more complex workflows.**

[**Storing secrets**](https://docs.github.com/en/actions/writing-workflows/about-workflows#storing-secrets)

**If your workflows use sensitive data, such as passwords or certificates, you can save these in GitHub as *secrets* and then use them in your workflows as environment variables. This means that you will be able to create and share workflows without having to embed sensitive values directly in the workflow's YAML source.**

**This example job demonstrates how to reference an existing secret as an environment variable, and send it as a parameter to an example command.**

**jobs:**

**example-job:**

**runs-on: ubuntu-latest**

**steps:**

**- name: Retrieve secret**

**env:**

**super\_secret: ${{ secrets.SUPERSECRET }}**

**run: |**

**example-command "$super\_secret"**

**For more information, see "**[**Using secrets in GitHub Actions**](https://docs.github.com/en/actions/security-guides/using-secrets-in-github-actions)**."**

[**Creating dependent jobs**](https://docs.github.com/en/actions/writing-workflows/about-workflows#creating-dependent-jobs)

**By default, the jobs in your workflow all run in parallel at the same time. If you have a job that must only run after another job has completed, you can use the needs keyword to create this dependency. If one of the jobs fails, all dependent jobs are skipped; however, if you need the jobs to continue, you can define this using the if conditional statement.**

**In this example, the setup, build, and test jobs run in series, with build and test being dependent on the successful completion of the job that precedes them:**

**jobs:**

**setup:**

**runs-on: ubuntu-latest**

**steps:**

**- run: ./setup\_server.sh**

**build:**

**needs: setup**

**runs-on: ubuntu-latest**

**steps:**

**- run: ./build\_server.sh**

**test:**

**needs: build**

**runs-on: ubuntu-latest**

**steps:**

**- run: ./test\_server.sh**

**For more information, see "**[**Using jobs in a workflow**](https://docs.github.com/en/actions/using-jobs/using-jobs-in-a-workflow#defining-prerequisite-jobs)**."**

[**Reusing workflows**](https://docs.github.com/en/actions/writing-workflows/about-workflows#reusing-workflows)

You can call one workflow from within another workflow. This allows you to reuse workflows, avoiding duplication and making your workflows easier to maintain. For more information, see "[Reusing workflows](https://docs.github.com/en/actions/using-workflows/reusing-workflows)."

[**Security hardening for workflows**](https://docs.github.com/en/actions/writing-workflows/about-workflows#security-hardening-for-workflows)

GitHub provides security features that you can use to increase the security of your workflows. You can use GitHub's built-in features to ensure you are notified about vulnerabilities in the actions you consume, or to automate the process of keeping the actions in your workflows up to date. For more information, see "[Using GitHub's security features to secure your use of GitHub Actions](https://docs.github.com/en/actions/security-guides/using-githubs-security-features-to-secure-your-use-of-github-actions)."