GitHub Actions CI/CD workflows

START HERE

`on: push`
Run on every push

`needs: <job>`
Run after another job

`runs-on: ubuntu-latest`
Pick a runner

`if: github.ref == 'refs/heads/main'`
Run conditionally

`uses: actions/checkout@v4`
Get the code

`${{ secrets.<NAME> }}`
Read a secret

`run: <command>`
Run a shell command

`strategy.matrix`
Fan out over versions

CONCEPTS

Workflows live in .github/workflows/*.yml; each file holds jobs, each job runs steps on one fresh runner.

Workflow one YAML file, triggered by events
- **Job** own runner, parallel unless needs
  - **Step** run: command or uses: action

Jobs do not share a filesystem: pass data with outputs or artifacts.

Pin third-party actions to a full commit SHA; tags can be moved.

TRIGGERS

`on: push`
Run on pushes

`on: pull_request`
Run on pull requests

`on: [push, pull_request]`
Several events at once

`on.push.branches: [main, 'release/**']`
Only these branches

`on.push.tags: ['v*']`
Run on version tags

`on.<event>.paths: ['src/**']`
Only when these files change
A skipped workflow leaves a required check pending.

`on.pull_request.types: [opened, synchronize]`
Limit PR activity types

`on.workflow_dispatch`
Allow manual runs
Only fires from the default branch.

`on.workflow_dispatch.inputs.<name>`
Manual run parameters

`on.schedule: - cron: '0 3 * * 1'`
Run on a cron schedule (UTC)

`on.workflow_call`
Make workflow reusable

`on.pull_request_target`
PR event with base-branch secrets
Never check out and run untrusted PR code with it.

JOBS

`jobs.<id>.runs-on: ubuntu-latest`
Choose the runner

`needs: [<job>, <job>]`
Wait for other jobs

JOBS (cont.)

`if: github.event_name == 'push'`
Condition on a job or step

`if: always()`
Run even after failure

`if: failure()`
Only after an earlier failure

`timeout-minutes: <n>`
Cancel a job or step after n min
Job default is 360.

`continue-on-error: true`
Do not fail the run on error

`environment: <name>`
Use environment secrets and approvals

`jobs.<id>.env`
Job-wide environment variables

STEPS

`- uses: <owner>/<repo>@<ref>`
Run an action

`with: { <input>: <value> }`
Pass inputs to an action

`- run: <command>`
Run a shell command

`run: |`
Multi-line script

`- name: <text>`
Label a step in the log

`id: <step_id>`
Name a step to read its outputs

`env: { <VAR>: <value> }`
Environment for a step

`shell: bash`
Choose the shell

`working-directory: <dir>`
Run a step elsewhere

MATRIX & CONTAINERS

`strategy.matrix.<key>: [a, b]`
One job per value

`${{ matrix.<key> }}`
Read the current value

`matrix.include: [{...}]`
Add or extend combinations

`matrix.exclude: [{...}]`
Drop combinations

`strategy.fail-fast: false`
Let other legs finish
Default is true: one failure cancels the rest.

`container: <image>`
Run the job in a container

`services.<id>.image: <image>`
Sidecar such as a database

MATRIX & CONTAINERS (cont.)

`services.<id>.options: --health-cmd ...`
Docker flags, health checks

EXPRESSIONS & CONTEXTS

`${{ <expression> }}`
Evaluate an expression

`github.event_name`
Event that triggered the run

`github.ref / github.ref_name`
Full ref / short branch or tag

`github.sha`
Commit SHA of the run

`vars.<NAME>`
Configuration variable

`needs.<job>.outputs.<name>`
Output of a needed job

`steps.<id>.outputs.<name>`
Output of an earlier step

`runner.os`
Runner OS (Linux, Windows, macOS)

`contains(<a>, <b>)`
Substring or array membership

`startsWith(<s>, <prefix>)`
String prefix test

`hashFiles('**/package-lock.json')`
Hash of files, e.g. cache key

`toJSON(<v>) / fromJSON(<s>)`
Convert to and from JSON
fromJSON on an output builds a dynamic matrix.

`success() / failure() / cancelled()`
Status check functions

DATA & SECRETS

`secrets.<NAME>`
Read a repository secret

`secrets.GITHUB_TOKEN`
Automatic per-run token

`echo "k=v" >> "$GITHUB_OUTPUT"`
Set a step output

`echo "K=v" >> "$GITHUB_ENV"`
Set env for later steps

`echo "<dir>" >> "$GITHUB_PATH"`
Prepend to PATH for later steps

`jobs.<id>.outputs.<name>: ${{ steps.<id>.outputs.<name> }}`
Expose step output to other jobs

DATA & SECRETS (cont.)

`on.workflow_call.outputs`
Return values from a reusable workflow

PERMISSIONS & CONCURRENCY

`permissions: contents: read`
Least-privilege GITHUB_TOKEN
Naming any scope sets the rest to none.

`permissions: id-token: write`
Allow OIDC to cloud providers

`concurrency.group: <name>`
One run per group at a time

`concurrency.cancel-in-progress: true`
Cancel superseded runs

REUSE, CACHE & ARTIFACTS

`uses: actions/cache@v4`
Cache dependencies

`uses: actions/upload-artifact@v4`
Keep files from a run

RECIPES

Minimal CI

```yaml
# .github/workflows/ci.yml
name: CI
on: [push, pull_request]
jobs:
  test:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v4
      - run: make test
```

Test on a matrix

```yaml
strategy:
  fail-fast: false
  matrix:
    os: [ubuntu-latest, windows-latest]
    node: [20, 22]
runs-on: ${{ matrix.os }}
```

Output from step to job

```yaml
- id: v
  run: echo "ver=1.2" >> "$GITHUB_OUTPUT"
- run: echo "${{ steps.v.outputs.ver }}"
```

Cancel stale runs

```yaml
concurrency:
  group: ${{ github.workflow }}-${{ github.ref }}
  cancel-in-progress: true
```

Deploy only from main

```yaml
deploy:
  needs: test
  if: github.ref == 'refs/heads/main'
  environment: production
```