GitHub Actions Quick Reference: Workflow & Action Attributes

Purpose

A zero-fluff reference for the core attributes you will use when designing GitHub Actions `workflows` and `actions`. Each snippet is production-friendly and highlights required vs optional bits.

1 Workflow Skeleton

File: `.github/workflows/ci.yml`

```yaml
name: CI

on:
  push:
    branches: [ main ]
    paths-ignore: [ "docs/**" ]
  pull_request:
    branches: [ main ]
    types: [opened, synchronize, reopened]
  workflow_dispatch:
    inputs:
      env:
        description: "Target environment"
        required: true
        type: choice
        options: [dev, staging, prod]
  schedule:
    - cron: "0 9 * * 1"   # Runs 09:00 UTC every Monday

jobs:
  build:                    # job ID (starts with letter or underscore)
    runs-on: ubuntu-latest # runner image
    permissions:
      contents: read
      id-token: write
    env:
      NODE_ENV: test
    steps:
      - name: Checkout
        uses: actions/checkout@v4

      - name: Setup Node
        uses: actions/setup-node@v4
        with:
          node-version: "22"

      - name: Install deps
        run: npm ci
```

```yaml
      - name: Run tests
        run: npm test -- --ci

      - name: Upload test reports
        uses: actions/upload-artifact@v4
        with:
          name: junit
          path: reports/junit.xml
```

Key attributes

- `name` (optional): Human-friendly workflow name; if omitted, GitHub derives one from the file path.
- `on` (required): Events that trigger the workflow. Common: `push`, `pull_request`, `workflow_dispatch`, `schedule`. Cron is **UTC**.
- `jobs` (required): Map of job identifiers to job definitions. Each workflow needs at least one job.

2 Jobs

- **Identifier rules**: start with letter or underscore; only letters, numbers, dashes, underscores.
- `runs-on`: Virtual environment for the job. Hosted: `ubuntu-latest`, `windows-latest`, `macos-latest`. You can also use `self-hosted`.
- `needs`: Express job dependencies (fan-in/fan-out).
- `strategy.matrix`: Run a job across a cartesian set of parameters.
- `concurrency`: Prevent overlapping runs for a given key.

Example with dependencies, matrix, and concurrency:

```yaml
jobs:
  lint:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v4
      - run: npm ci && npm run lint

  test:
    needs: [lint]
    runs-on: ubuntu-latest
    strategy:
      fail-fast: false
      matrix:
        node: [18, 20, 22]
        os: [ubuntu-latest, windows-latest]
    runs-on: ${{ matrix.os }}
    concurrency:
      group: tests-${{ github.ref }}-${{ matrix.os }}-node${{ matrix.node }}
      cancel-in-progress: true
    steps:
      - uses: actions/checkout@v4
      - uses: actions/setup-node@v4
        with: { node-version: ${{ matrix.node }} }
      - run: npm ci
      - run: npm test -- --ci
```

3 Steps

- Each `step` runs in its own process but shares the job's workspace.
- `name` (optional): Label for the step.
- `uses`: Reference to an *action* (prebuilt logic).
- `run`: Shell commands executed on the runner.
- `shell`, `working-directory`, `env`: Per-step customizations.

Example mixing `uses` and `run`:

```yaml
steps:
  - name: Checkout
    uses: actions/checkout@v4

  - name: Use a Docker action
    uses: docker://alpine:3.20
    with:
      args: sh -lc 'echo "Hello from Docker"'

  - name: Build
    run: |
      make clean
      make -j"$(nproc)"

  - name: Conditional step
    if: ${{ success() && github.event_name == 'push' }}
    run: echo "Ran on push and previous steps passed"
```

4 Referencing Actions with `uses`

- Public repo action: `owner/repo@ref` where `ref` is a tag, branch, or *preferably* a commit SHA.
- Local action: path relative to workflow file, e.g., `./.github/actions/my-action`.
- Container registry action: `docker://image:tag`.
- **Pin versions**: For supply chain safety, pin to a full commit SHA when possible.

Examples:

```yaml
- uses: actions/checkout@3df4f2a...   # pinned to commit
- uses: my-org/my-action@v2           # pinned to tag
- uses: ./.github/actions/lint        # local action
- uses: docker://ghcr.io/org/tool:1.2 # container action
```

5 Triggers via on

Common patterns:

```yaml
on:
  push:            # most common, fires on new commits
    branches: [ main, release/** ]
    paths: [ "src/**", "!**/*.md" ]

  pull_request:    # PR lifecycle
    types: [opened, synchronize, reopened, ready_for_review]

  workflow_dispatch:  # manual with inputs
    inputs:
      ref:
        description: "Git ref (branch or tag)"
        required: false
        default: main

  schedule:
    - cron: "30 2 * * *" # 02:30 UTC daily
```

Notes

- **Cron timezone is UTC**. Convert carefully if you expect local time.
- `repository_dispatch` and webhooks can kick off workflows from external systems.

6 Permissions & Security Basics

- Use least privilege via top-level or job-level `permissions`.
- Prefer **OIDC** (`id-token: write`) to cloud secrets where supported.
- Never echo secrets; use `secrets.MY_SECRET` and mask output if needed.

```yaml
permissions:
  contents: read
  pull-requests: write
  id-token: write   # for OIDC federation
```

7 Reusable Workflows vs Actions

Reusable workflow (workflow_call)

Wrap a whole workflow and call it from others.

```yaml
# .github/workflows/reusable-build.yml
name: Reusable Build
on:
  workflow_call:
    inputs:
      node:
        required: true
        type: string

jobs:
  build:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v4
      - uses: actions/setup-node@v4
        with: { node-version: ${{ inputs.node }} }
      - run: npm ci && npm test

# caller workflow
on: [push]
jobs:
  call-build:
    uses: org/repo/.github/workflows/reusable-build.yml@v1
    with:
      node: "22"
```

Action (composite) basics

Great for bundling repeatable step logic. **File:** `.github/actions/hello/action.yml`

```yaml
name: "Hello"
description: "Say hello"
inputs:
  who:
    description: "Name"
    required: true
runs:
  using: "composite"
  steps:
    - run: echo "Hello, ${{ inputs.who }}!"
      shell: bash
```

8 Job Outputs & Conditionals

```yaml
jobs:
  gather:
    runs-on: ubuntu-latest
    outputs:
      sha: ${{ steps.pick.outputs.sha }}
    steps:
      - id: pick
        run: echo "sha=${GITHUB_SHA}" >> "$GITHUB_OUTPUT"

  use:
    needs: [gather]
    runs-on: ubuntu-latest
    if: ${{ needs.gather.outputs.sha != '' }}
    steps:
      - run: echo "Downstream got ${{ needs.gather.outputs.sha }}"
```

9 Self-Hosted Runners (Heads-Up)

- Use when you need special software or hardware.
- Lock down network egress, harden the host, and rotate runner tokens.
- Tag runners and target with `runs-on: [self-hosted, linux, gpu]` etc.

10 Common Pitfalls

- **Missing `on`**: Workflow will never trigger.
- **Wrong job IDs**: Must match identifier rules; avoid spaces and special chars.
- **Cron surprises**: Cron uses **UTC**, not local time.
- **Unpinned actions**: Prefer commit SHAs for security and reproducibility.
- **Secrets in logs**: Use `secrets.*` and avoid printing them.

Build Notes

Ensure Pygments is installed. Example compile:

```
# One of the following:
xelatex -shell-escape file.tex
lualatex -shell-escape file.tex
```