

# GitHub Actions — Practical Cheat Sheet

**Scope.** A rapid, no-fluff reference for: (1) workflow placement & triggers, (2) reusable workflows, (3) authoring custom actions, (4) job summaries, (5) CODEOWNERS, and (6) org workflow templates—plus common gotchas and a setup checklist.

## 1 Workflows & Triggers (Placement That *Actually* Works)

- Put workflow files in `.github/workflows/` (lower-case, plural). Extensions: `.yaml` or `.yml`.
- Many events (e.g., `pull_request`, `issues`, `release`, `schedule`, UI `workflow_dispatch`) are only recognized when the workflow file exists on the repository's *default branch*.
- File name does not matter; the `name:` field sets the display name in the UI.

```
1 # .github/workflows/ci.yml
2 name: CI
3 on:
4   push:
5     branches: [ main ]
6   workflow_dispatch:
7 jobs:
8   build:
9     runs-on: ubuntu-latest
10    steps:
11      - uses: actions/checkout@v4
12      - run: echo "Build stuff"
```

### Event filters that save time.

- Use `paths/paths-ignore` to scope when a workflow runs.
- For `pull_request`, prefer `pull_request_target` only when you *must* run with base-branch permissions (be mindful of security).

## 2 Reusable Workflows (Calling & Inputs)

- Reusable workflows are also stored under `.github/workflows/`.
- Call with `jobs.<id>.uses: <owner>/<repo>/.github/workflows/<file>@<ref>`.
- Pass parameters via `with:` and secrets via `secrets:`.

```
1 # .github/workflows/reuse-caller.yml
2 name: Reuse CI
3 on: workflow_dispatch
4 jobs:
5   call-reusable:
6     uses: your-org/your-repo/.github/workflows/ci.yml@v1
7     with:
8       some-input: "value"
9     secrets:
10      TOKEN: ${ secrets.MY_TOKEN }
```

**Tip.** Version reusable workflows with moving tags (v1) plus immutable releases (v1.2.3). Update the moving tag on each release to provide stability with opt-in upgrades.

## 3 Authoring Custom Actions (JavaScript or Docker)

- Prefer one published action per repo with definition at repo root as `action.yml` (or `action.yaml`).
- You *can* keep multiple actions in subfolders, but that repository cannot be published to the Marketplace as a single action.
- Set `runs.using` to `node20` for JavaScript actions; point `main` at your compiled entry (e.g., `dist/index.js`).

```
1 # action.yml (JavaScript action)
2 name: "Hello Action"
3 description: "Greets and exits"
4 runs:
5   using: "node20"
6   main: "dist/index.js"
7 inputs:
8   who:
9     description: "Who to greet"
10    default: "world"
```

Using your action:

```
1 - uses: your-org/hello-action@v1
2   with:
3     who: "Jordan"
```

**Release hygiene.** Create a release (e.g., v1.2.3), then move the major tag (e.g., v1) to that commit. Consumers can pin either `@v1` or `@v1.2.3`.

## 4 Job Summaries (Fast, Pretty Reports)

Write Markdown to `$GITHUB_STEP_SUMMARY` to show a per-job summary in the UI.

- One summary file per job, up to 20 writes, about 1 MB total content.
- Supports tables, code fences, and Mermaid diagrams in GitHub Markdown.

```
1 # step: Summarize test results
2 echo "## Test Summary" >> "$GITHUB_STEP_SUMMARY"
3 echo "" >> "$GITHUB_STEP_SUMMARY"
4 echo "| Metric | Value |" >> "$GITHUB_STEP_SUMMARY"
5 echo "|---|---:|" >> "$GITHUB_STEP_SUMMARY"
6 echo "| Total | 120 |" >> "$GITHUB_STEP_SUMMARY"
7 echo "| Passed | 118 |" >> "$GITHUB_STEP_SUMMARY"
8 echo "| Failed | 2 |" >> "$GITHUB_STEP_SUMMARY"
```

## 5 CODEOWNERS (Code Owner Reviews That Stick)

- File must be named `CODEOWNERS` (exact name). Valid locations: repo root, `.github/`, or `docs/`.
- The file must exist on the *base branch* for a PR to apply owners. Combine with Branch Protection: “Require review from Code Owners.”
- Pattern order matters; the *first* match wins.
- Each pattern must list at least one owner (users or teams); owners must have *write* access to count as code owners.
- No negation (!) and no bracket character classes in patterns.

```
1 # CODEOWNERS (place at /.github/CODEOWNERS or at repo root)
2 # Comments begin with '#'
3 /docs/           @org/docs-team
4 /apps/           @octocat
5 /apps/secret/    @org/security-reviewers
```

**Enforcement.** In *Branch protection rules*, enable: Require a pull request before merging → Require review from Code Owners. Owners cannot auto-approve their own PRs unless allowed by policy.

## 6 Org Workflow Templates (Show Up in “New Workflow”)

To offer curated starter workflows across the org:

1. Create a *public* org repository named `.github`.
2. Add templates under `workflow-templates/`.
3. Each template needs a workflow file and a matching `properties/*.properties.json`.
4. Icon can be an Octicon name or an SVG in the folder (reference by filename without `.svg`).
5. Templates are discovered from the default branch.

```
1 # .github/workflow-templates/node-ci.yml
2 name: Node CI
3 on: [push]
4 jobs:
5   build:
6     runs-on: ubuntu-latest
7     steps:
8       - uses: actions/checkout@v4
9       - uses: actions/setup-node@v4
10         with: { node-version: 20 }
11       - run: npm ci
12       - run: npm test --silent
```

```
1 // .github/workflow-templates/properties/node-ci.properties.json
2 {
3   "name": "Node CI (Org Standard)",
4   "description": "Lint, test, and build using our org baseline.",
5   "iconName": "repo-push",
6   "categories": ["JavaScript"]
7 }
```

## 7 Security, Permissions, Caching (Quick Wins)

- **GITHUB\_TOKEN permissions:** set least privilege at the workflow or job level.

```
1 permissions:
2   contents: read
3   pull-requests: write # only if you need to comment/update PRs
```

- **Concurrency:** cancel superseded builds to save minutes.

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

- **Caching:** pair cache keys with a fallback to avoid cache misses.

```
1 - uses: actions/cache@v4
2   with:
3     path: ~/.npm
4     key: npm-${runner.os}-${hashFiles('**/package-lock.json')}
5     restore-keys: |
6       npm-${runner.os}-
```

## 8 Gotchas & Troubleshooting

- **Default branch rule:** Schedules and many UI-driven triggers use the workflow from the default branch.
- **Path filters:** When using `paths`, remember that a workflow still appears in the run list even if all jobs are skipped by conditions.
- **Reusable workflows:** Inputs are strings by default; coerce types in the called workflow if needed.
- **Marketplace pinning:** Pin third-party actions to a tag and, ideally, a commit SHA for supply-chain safety.
- **Matrix limits:** Large matrices can hit concurrency or minutes limits—chunk by `include/exclude`.

## 9 New Repository Setup — Mini Checklist

- Create `.github/workflows/ci.yml` with `push` and `workflow_dispatch`.
- Set **Branch protection:** require PRs, require status checks, and require review from Code Owners.
- Add CODEOWNERS at repo root or `.github/`.
- Lock down GITHUB\_TOKEN permissions to least privilege.
- Configure cache for your package manager and test runner.
- (Org-wide) Keep curated templates in the public `.github` repo under `workflow-templates/`.

This cheat sheet is intended as a fast operational reference. For deeper context, consult the official GitHub Actions documentation for events, expressions, and security guidance.