

GitHub Actions Practical Workbook

Visual handwritten notes - original labs and solutions

Understand

Clear concepts and practical decisions

Visualize

Diagrams that show how tools connect

Practice

Original examples and evidence checks

Remember

Exercises and a separate quiz answer key

18 topics - original Release Desk project - explained answer key

From a first workflow to CI/CD review

Practical edition 1.0 | 29 October 2026

Course map I

Read concepts, try the task, then compare the worked solution.

Topic	Concept / practice / solution
01. Workflow, job, step and runner	6 / 7 / 42
02. Events and branch filters	8 / 9 / 42
03. Checkout and reproducible installation	10 / 11 / 43
04. YAML, shell and expressions	12 / 13 / 43
05. Matrix testing with clear scope	14 / 15 / 44
06. Jobs, dependencies and outputs	16 / 17 / 44
07. Cache and artifact are different tools	18 / 19 / 45
08. Artifacts and evidence retention	20 / 21 / 45
09. Permissions and the repository token	22 / 23 / 46

A consistent learning loop

- + Read the concept and predict the command result before running the walkthrough.
- + Attempt the practical exercise without opening solutions/. Capture actual results.
- + Compare your reasoning with the worked solution. Finish the mini quiz before checking the final answer key.

LAB RULE. Topics are independent fixtures unless a prerequisite is stated. Reuse the supplied package structure; do not run destructive Git examples in your own project.

Worked practical solutions begin on page 42. The capstone, references and quiz key follow the lesson section. PDF bookmarks link directly to each topic and appendix.

Course map 2

Read concepts, try the task, then compare the worked solution.

Topic	Concept / practice / solution
10. Secrets and untrusted pull requests	24 / 25 / 46
11. Service containers and health gates	26 / 27 / 47
12. Docker build and publish boundaries	28 / 29 / 47
13. Reusable workflows and contracts	30 / 31 / 48
14. Concurrency and cancellation policy	32 / 33 / 48
15. Environments, OIDC and approval	34 / 35 / 49
16. Debug a failed pipeline with evidence	36 / 37 / 49
17. Release rehearsal and rollback record	38 / 39 / 50
18. Version pins and safe maintenance	40 / 41 / 50

A consistent learning loop

- + Read the concept and predict the command result before running the walkthrough.
- + Attempt the practical exercise without opening solutions/. Capture actual results.
- + Compare your reasoning with the worked solution. Finish the mini quiz before checking the final answer key.

LAB RULE. Topics are independent fixtures unless a prerequisite is stated. Reuse the supplied package structure; do not run destructive Git examples in your own project.

Worked practical solutions begin on page 42. The capstone, references and quiz key follow the lesson section. PDF bookmarks link directly to each topic and appendix.

Your local learning environment

Keep the full companion folder beside this PDF.

Item	Requirement / scope
Companion	devops-learning-companion/ from the matching complete ZIP
App runtime	Node.js 22 or 24; no third-party npm dependencies
Git labs	Git and Bash; fresh disposable fixtures generated by tools/verify_labs.py
Linux target	Ubuntu 24.04 LTS VM for administration; Bash-only tasks also checked on Git Bash
Actions target	Your own practice repository; workflows copied at its root
Privileges	No root needed for portable/Git/app checks. VM provisioning is a separate deliberate task.

Start with the application

```

● ● ● TERMINAL / APP BASELINE
cd devops-learning-companion/app
npm ci --ignore-scripts
npm test
npm run evidence

```

4 lines

The service uses only built-in Node modules. `npm ci` checks the included lockfile; the `test` command runs 12 reference checks. `npm run evidence` writes the actual runtime, exit code and TAP report under `app/evidence/`.

Use the correct shell

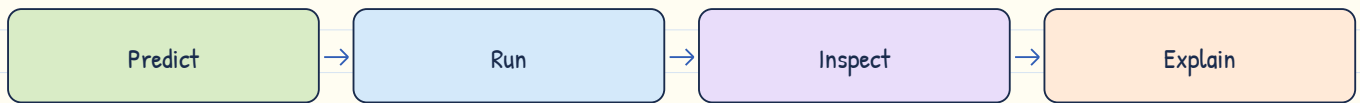
Bash examples use `printf`, pipes and Unix-style quoting. Run them in Bash, not directly in PowerShell. Open Git Bash on Windows for portable labs; use the documented Ubuntu VM for users, permissions, packages, networking and services. Run each lesson in its own generated fixture.

PROJECT SCOPE. The sample service intentionally implements only `GET /health` and `GET /release`. It is a small operations fixture, not a production authentication, persistence or TLS implementation.

A repeatable study method

Build understanding through predictions, observations and explanations.

DIAGRAM / FOLLOW THE BOUNDARY



What a completed exercise contains

- + Task: the exact behavior you intend to demonstrate, with a safe starting state.
- + Expected output: stable content or status to compare; commit IDs, timing and platform text may vary.
- + Evidence: command, observed output, exit code and relevant file or HTTP state.
- + Failure case: show how a wrong input or missing precondition is rejected.
- + Cleanup: stop the listener and keep only the fixtures/evidence you intend to retain.

Find the companion material

Path	Purpose
actions/lessons/	Exercise briefs; one numbered file per topic
actions/walkthroughs/	Concept demonstrations; Actions files are labelled fragments
actions/solutions/	Original worked practical solutions
app/	Shared original service, 12 tests and local promotion rehearsal
tools/verify_labs.py	Isolated Git and portable Bash verification
actions/HOSTED-VALIDATION.md	Checklist to collect actual hosted evidence

Practical solutions are grouped after all lessons so you can attempt tasks first. Quiz answers are placed at the very end. There are 36 questions per book; award one point for a correct explanation and revise any answer that omits the stated limitation.

Prepare an exercise without its solution

```

FROM THE COMPANION ROOT

python tools/prepare_lab.py git 06 --walkthrough
# Use linux instead of git for a shell lesson.
# Change directory to the printed student path.
  
```

3 lines

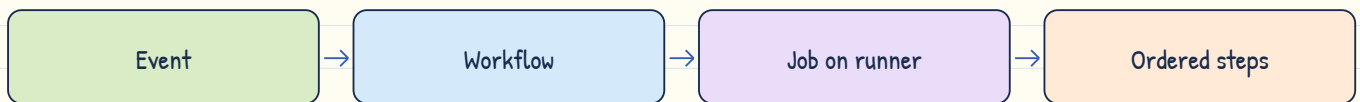
Each call creates a new persistent fixture and runs only the walkthrough. Attempt the task in the printed directory. Omit `--walkthrough` to practise the demonstration yourself. VM-only commands are never automatically run.

Workflow, job, step and runner

Release Desk: one tool decision, one observable result.

A workflow responds to an event. A job executes on a runner, and its steps share that job filesystem and environment. Different jobs normally start with separate workspaces. Release Desk separates testing from release decisions so a successful shell command cannot silently promote code. Begin with the complete companion workflow; fragments in later lessons show only the named portion to change.

DIAGRAM / FOLLOW THE BOUNDARY



Walk through the behavior

```
name: Release Desk check
on: [workflow_dispatch]
permissions:
  contents: read
jobs:
  inspect:
    runs-on: ubuntu-24.04
    steps:
      - run: printf 'workflow started\n'
```

Expected outcome

In your own GitHub repository, a manual run would create one inspect job with one shell step. Local parsing alone does not execute a hosted run.

WHEN IT FAILS. If no workflow appears, confirm the file is under `.github/workflows/` with `.yml` or `.yaml` and that manual-trigger availability follows the repository default branch.

Companion: [actions/walkthroughs/01.txt](#) | Reference links in the source directory.

Lab: Workflow, job, step and runner

Try the task before reading the worked solution.

Your task

Add a second step that prints finished. Explain whether it shares files with the first step.

Acceptance and evidence

Checkpoint	What to record
Context	Use the complete workflows for execution; lesson fragments show insertion context
Local evidence	App tests and workflow static validation
Hosted evidence	Run URL, source SHA, actual runner/runtime and check results
Scope	No hosted execution or cloud deployment is claimed by the author baseline

Predict one failure

If no workflow appears, confirm the file is under `.github/workflows/` with `.yaml` or `.yml` and that manual-trigger availability follows the repository default branch.

Mini quiz

Q1. Does each step get a new hosted VM?

Q2. Does valid YAML prove a workflow ran?

Try both questions before checking the quiz answer key at the end of this workbook.

Your evidence

Command / change: _____

Observed result and exit status: _____

Remaining question: _____

Practical model: page 42. Quiz key: page 62.