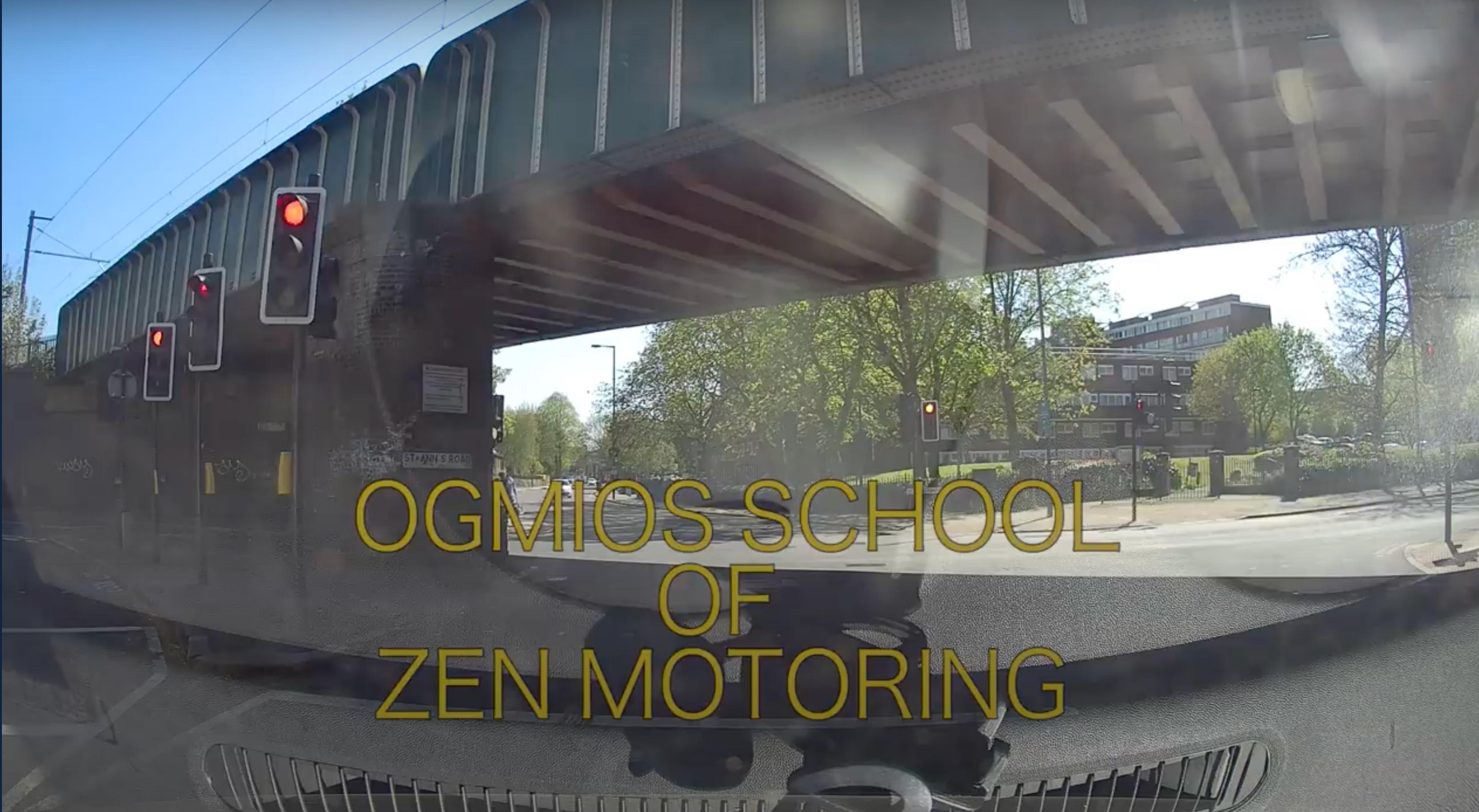


The Zen School of GitHub Actions

Jonathan Ruckwood
@jonruckwood

Lead Consultant, OpenCredo



@ogmiosmusic

“the beat-poet Attenborough”

 @jonruckwood

GitHub Actions

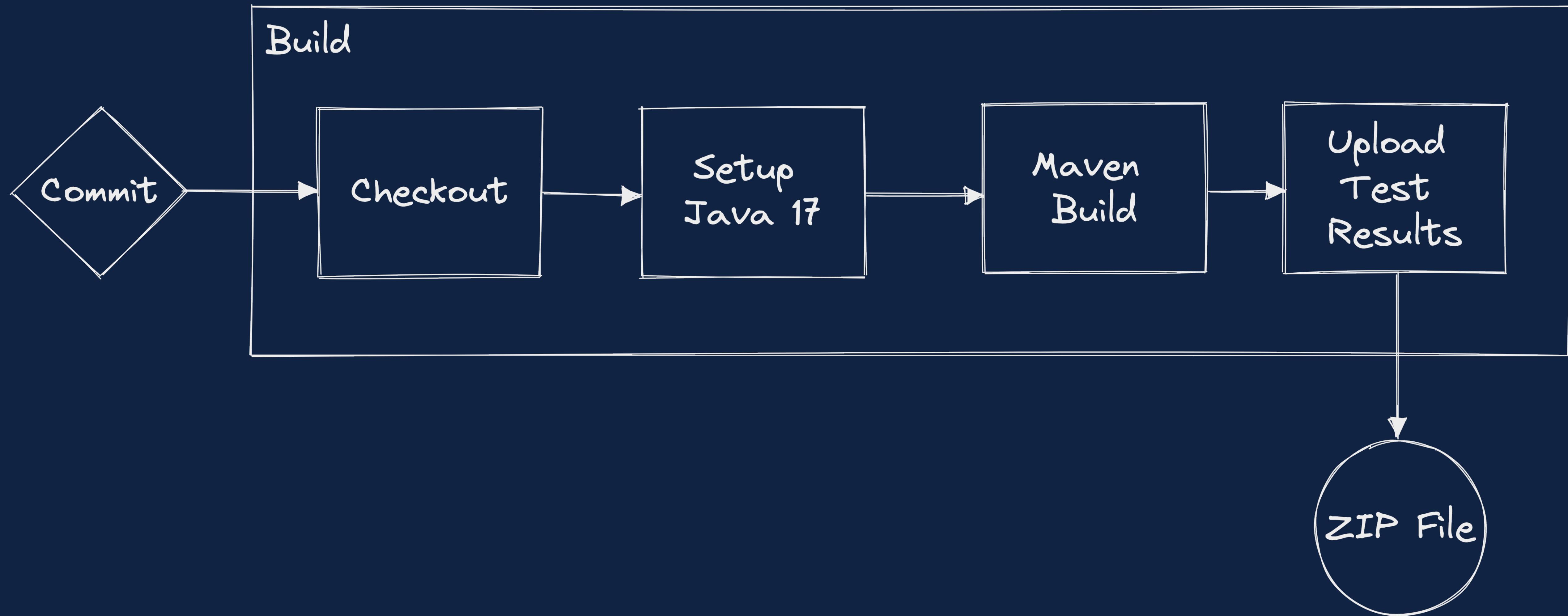
- Frustrating, unpredictable and unforgiving

A familiar experience...

Why?

- Low Friction
- Benefits smaller teams
- Thriving eco-system
- Well documented

Build a Maven Project



Triggering Events

Jobs

Steps

```
name: Build

on:
  push:
    branches:
      - main

  workflow_dispatch:

jobs:
  build:
    runs-on: ubuntu-20.04

    steps:
      - uses: actions/checkout@v3

      - uses: actions/setup-java@v3
        with:
          java-version: 17
          distribution: 'zulu'
          cache: 'maven'

      - run: mvn package --file pom.xml

      - uses: actions/upload-artifact@v2
        if: ${{ failure() }}
        with:
          name: surefire-reports
          path: |
            **/target/surefire-reports/
```

.github/workflows/build.yaml

A screenshot of a GitHub Actions build summary page. The URL in the browser is `github.com/jon-ruckwood/devoxx-2022-simple-workflow`. The page shows a successful build (#5) triggered manually 2 days ago by user `jon-ruckwood` (commit `464a383`). The build duration was 43 seconds with 1m billable time. The workflow file `build.yaml` contains a single job named `build` which completed successfully in 25 seconds.

Build Build #5

Manually triggered 2 days ago

Status: Success

Total duration: 43s

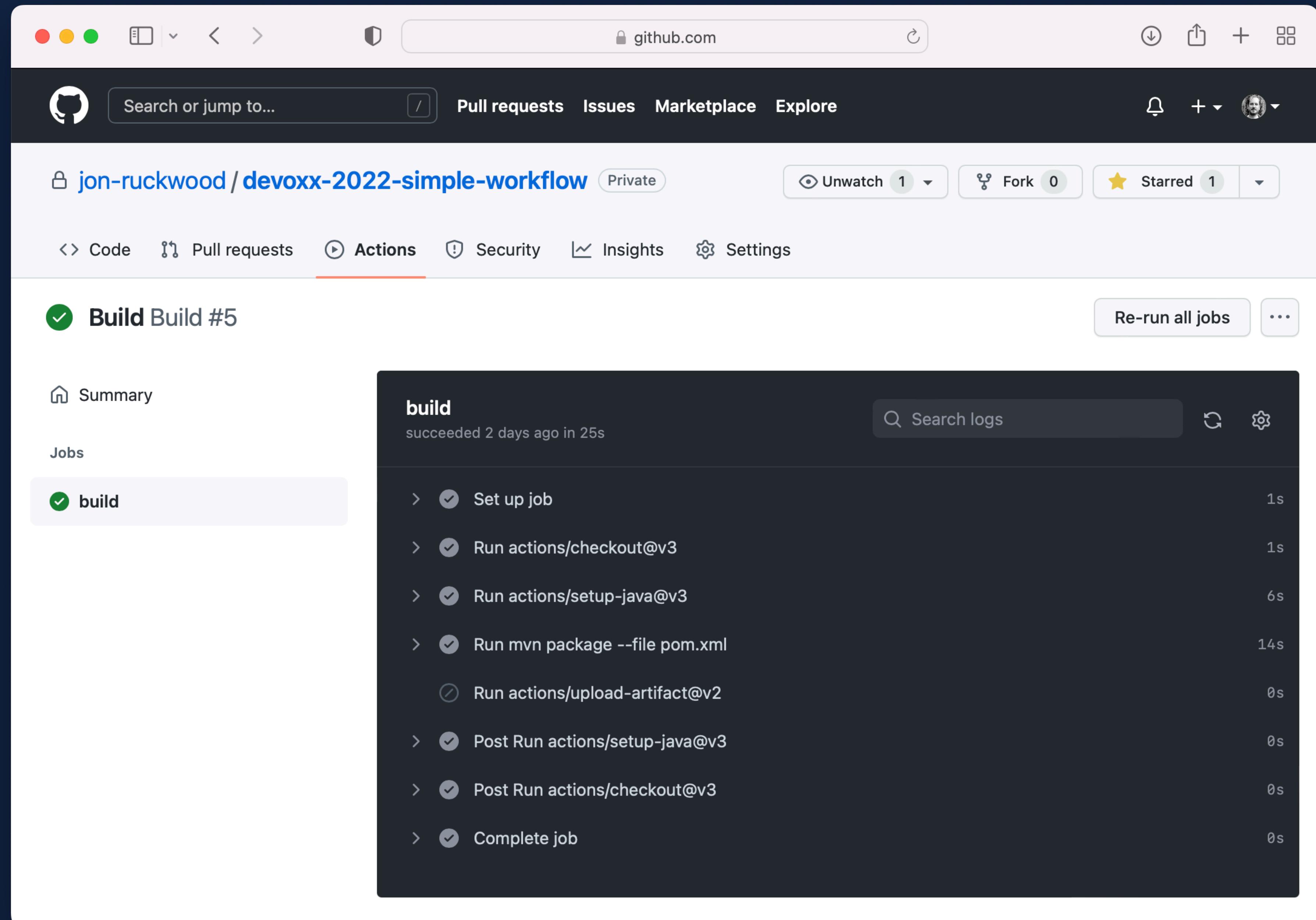
Billable time: 1m

Artifacts: -

build.yaml

on: workflow_dispatch

build 25s



A screenshot of a GitHub repository page showing the Actions tab. The repository is named `jon-ruckwood/devoxx-2022-simple-workflow`. The build log for Build #5 is displayed, showing a successful run completed 2 days ago in 25s. The log details the following steps:

- > ✓ Set up job 1s
- > ✓ Run actions/checkout@v3 1s
- > ✓ Run actions/setup-java@v3 6s
- > ✓ Run mvn package --file pom.xml 14s
- < ✓ Run actions/upload-artifact@v2 0s
- > ✓ Post Run actions/setup-java@v3 0s
- > ✓ Post Run actions/checkout@v3 0s
- > ✓ Complete job 0s

A screenshot of a GitHub Actions build log for a repository named "jon-ruckwood / devoxx-2022-simple-workflow". The build is labeled "Build #5" and has succeeded 2 days ago in 25s. The log shows the following steps:

- > Set up job (1s)
- > Run actions/checkout@v3 (1s)
- > Run actions/setup-java@v3 (6s)
- > Run mvn package --file pom.xml (14s)
 - 1 Run mvn package --file pom.xml
 - 6 [INFO] Scanning for projects...
 - 7 [INFO]
 - 8 [INFO] -----< net.selfdotlearn:devoxx-2022-simple-workflow >-----
 - 9 [INFO] Building devoxx-2022-simple-workflow 0.0.0-SNAPSHOT
 - 10 [INFO] -----[jar]-----
 - 11 [INFO]
 - 12 [INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ devoxx-2022-simple-workflow ---
 - 13 Warning: Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build

Tips

Caching

- Save on compute and your time
- Free!
- Up to 10GB per-repository
- Advanced usage: `actions/cache@v3`

Caching – Dependencies

```
- uses: actions/setup-java@v3
  with:
    java-version: 17
    distribution: 'zulu'
    cache: 'maven'
```

Caching – Docker

```
- uses: docker/setup-buildx-action@v1  
# --8<--  
  
- uses: docker/build-push-action@v2  
  with:  
    context: .  
    push: true  
    cache-from: type=gha  
    cache-to: type=gha,mode=max
```

- mode=max – export all the layers of all intermediate steps
- Most benefit from Dockerfiles with multiple layers
 - E.g. Spring Boot Maven Plugin

Limits

- Execution time of Jobs and Steps
 - timeout-minutes: 10
- Concurrency
 - Fixed key – concurrency: main
 - Variable – concurrency: ci-\${{ github.ref }}

Reuse & Composability

- Local action
 - uses: `.github/actions/my-first-action/action.yaml`
- Composite Action – an “action of actions”
- Reuse workflow from a different repository
 - `workflow_call`

Debug

- Add these secrets with value true
 - ACTIONS_RUNNER_DEBUG
 - ACTIONS_STEP_DEBUG

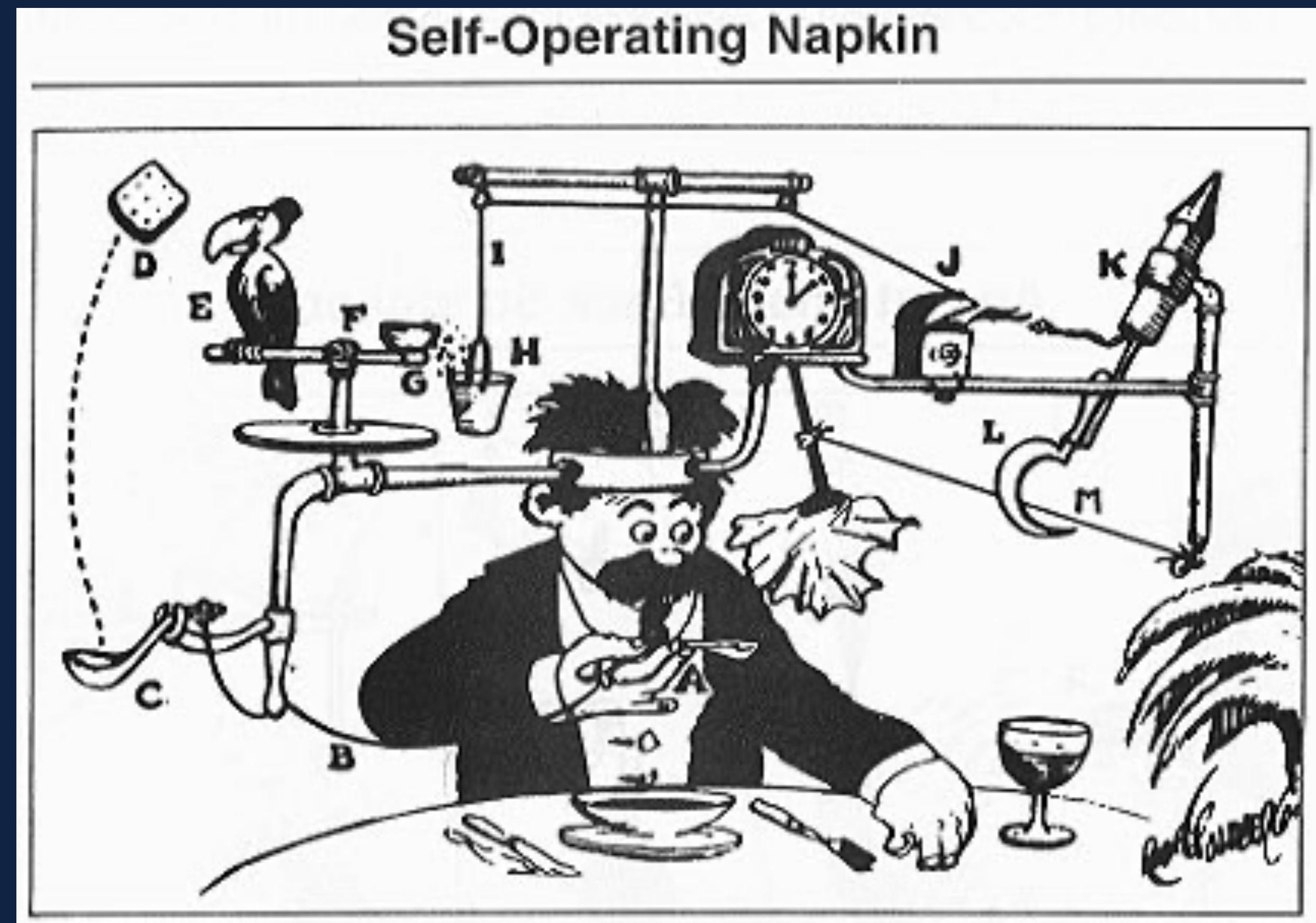
Act

- act – run your GitHub Actions locally
- GitHub: [nektos/act](https://github.com/nektos/act)

```
# Run the default (`push`) event:  
act  
  
# Run a specific event:  
act pull_request  
  
# Run a specific job:  
act -j test  
  
# Run in dry-run mode:  
act -n  
  
# Pass in a secret:  
act -s GITHUB_TOKEN=...
```

Design

Keep it simple



Keep it Lean

- Don't bake in too much!
- Use external scripts
- Lean on build tools, e.g. Spring Boot plugin

You need a backup plan

Backup Plan

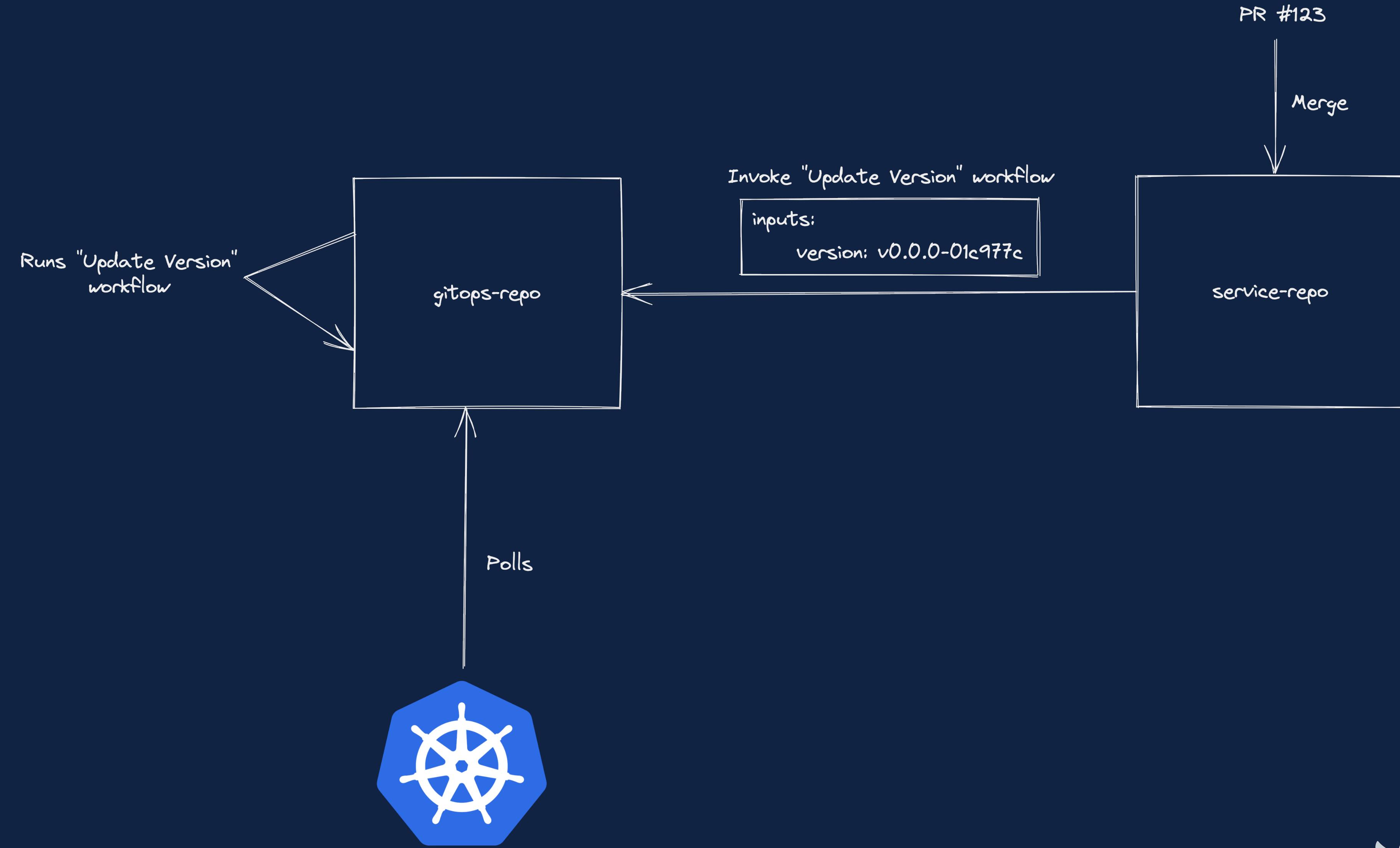
- Have a run-book for doing the essential things your workflow(s) but performed manually
- Printed out or documented on a wiki

Treat your repo as an API

Your repository as an API

- Treat the workflow as the *interface*
 - Ask a *repository* to perform action – don't do it yourself!
- Chaining workflows across repositories
 - `repository_dispatch` & `workflow_dispatch`

Example



Coupling – Be aware!

- Cross-repository invocation
- Shared actions
- Shared workflows

Security

Hardening

- Do not trust the Marketplace
- You **must** follow GitHub's security hardening guide
- Review your repository's settings
 - Don't allow workflows from fork PRs

Pin to Action to SHA

```
# ✗  
uses: peter-evans/create-pull-request@v4.0.2  
  
# ✓  
uses: peter-evans/create-pull-request@bd72e1b7922d417764d27d30768117ad7da78a0e # => v4.0.2
```

Permissions

- Limit GITHUB_TOKEN permissions
- Set default for repository and GitHub organisation

```
permissions:  
  actions: read  
  checks: read | write | none  
  contents: read | write | none  
  deployments: read | write | none  
  id-token: read | write | none  
  issues: read | write | none  
  discussions: read | write | none  
  packages: read | write | none  
  pages: read | write | none  
  pull-requests: read | write | none
```

```
# --8<--
```

```
permissions: read-all | write-all
```

```
# --8<--
```

```
permissions: {}
```

Workflow permissions

Choose the default permissions granted to the GITHUB_TOKEN when running workflows in this repository. You can specify more granular permissions in the workflow using YAML. [Learn more.](#)

Read and write permissions

Workflows have read and write permissions in the repository for all scopes.

Read repository contents permission

Workflows have read permissions in the repository for the contents scope only.

Consider

- Use a “Service Account” for things GITHUB_TOKEN cannot do
- Avoid long-lived secrets if interacting with things like AWS
 - OIDC provider for temporary session tokens

Security

- “How to use GitHub Actions with security in mind” – Rob Bos

Summary

- Understand there will be pain
- Don't sweat the small stuff
- Don't ignore security



devoxx-2022

About Me

- Twitter: [@jon-ruckwood](#)

The Zen School of GitHub Actions – Devoxx 2022

- Slides – [The Zen School of GitHub Actions](#) (PDF)

School of Zen Motoring

- Twitter: [@ogmiosmusic](#) / YouTube: [ogmiosmusic](#) / BBC iPlayer: [Zen Motoring](#)

Code Samples

- Simple workflow – [jon-ruckwood/devoxx-2022-simple-workflow](#)
- Cross-repository workflows:
 - Service repository – [jon-ruckwood/devoxx-2022-service](#)
 - GitOps repository – [jon-ruckwood/devoxx-2022-infrastructure](#)

Resources

- `act` run your GitHub Actions locally – [nekton/act](#)
- Security hardening for GitHub Actions – <https://docs.github.com/en/actions/security-guides/security-hardening-for-github-actions>
- How to use GitHub Actions with security in mind by Rob Bos – <https://www.youtube.com/watch?v=ErslcA7Nmc>

Thank you :-)

Jonathan Ruckwood
@jonruckwood

Lead Consultant, OpenCredo



GitHub: [jon-ruckwood/devoxx-2022](https://github.com/jon-ruckwood/devoxx-2022)