GitHub Actions

Overview

Practical Examples

GitHub Actions - Overview

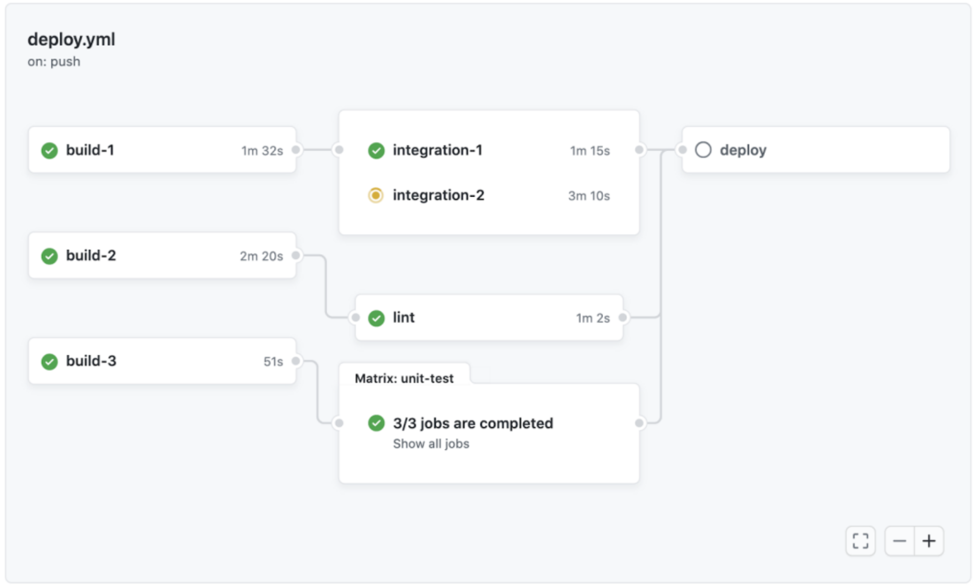
- GitHub Actions is a CI/CD automation platform
  - **CI** = Continuous Integration
  - **CD** = Continuous Delivery
  - Not just for building code: anything on GitHub can be automated
    - Pull requests, Issues, Pages, Deployment, …

- Core concept: “**workflow**”
  - Triggers: When X happens…
  - … do something

- Similar to other CI/CD platforms (usually called **pipelines**):
  - GitLab – Pipeline
  - Jenkins - Pipeline
  - Travis - Pipeline
  - Azure DevOps - Pipeline
  - AWS CodePipeline
  - Atlassian Bamboo

GitHub Actions - Terminology

- **GitHub Actions** is the name of the platform
- A **workflow** is a complete description of a set of work to perform when triggered (a.k.a. pipeline in other systems)
  - Workflows can be **reusable** (called from other repositories/workflows)
- A **job** is a unit of work inside a workflow
  - Jobs run in parallel, and can have dependencies
- Jobs consist of **steps**
- Steps can be shell scripts or **(reusable) actions**
  - Yes, this is confusing…
  - There is a large ecosystem of reusable actions



HBK
HOTTINGER BRÜEL & KJÆR

GitHub Actions – Simple Example

- [Example 1: https://github.com/dvnrrs/github-actions-demo/blob/main/.github/workflows/1-build-myprogram.yml]

GitHub Actions – Runners

- By default, GitHub cloud runners are used.
  - Examples:
    - `runs-on: ubuntu-latest`
    - `runs-on: ubuntu-latest-8-cores`
    - `runs-on: windows-2019`
    - `runs-on: macos-14-large`
  - Personal accounts (and paid ones) get a quota of public runner minutes.
  - “Large runners” must be enabled, and cost more.

- On-premises (self-hosted) runners can also be used.
  - `runs-on:`
    - `group: fusion-devops-linux`

- In both cases, Docker containers can be used.
  - `container: ghcr.io/blueberrydaq/buildroot-builder:2.9`
  - The repository must have permission to access the container.

GitHub Actions - Triggers

- push – Runs when a commit or tag is pushed to a repository.
  - `branches: [main, release/**]`
- pull_request - Runs when a pull request is created or modified.
  - `types: [opened, reopened]`
- workflow_dispatch – Runs when manually triggered from the GitHub UI.
- workflow_call – Runs when called from another workflow (reusable workflow).
- workflow_run – Runs when another workflow ends.
  - `workflows: [build]`
  - `types: [completed]`
- schedule – Runs at specific times.
  - `- cron: '30 5,17 * * *'`

GitHub Actions – Custom Actions

- There is a large ecosystem of reusable actions.
- You can create your own reusable action.
  - **Composite actions** combine existing actions in a sequence, like a workflow.
  - **JavaScript actions** are written using Node.js.
  - **Docker container actions** are implemented with Docker images.
- Actions can have **inputs** and **outputs** as well as **side effects**.
- [Example 2: https://github.com/dvnrrs/github-actions-demo/blob/main/.github/workflows/2-build-myprogram-with-reusable-action.yml]
- [JavaScript Example: https://github.com/blueberrydaq/git-version-action]

HBK
HOTTINGER BRÜEL & KJÆR

GitHub Actions – Reusable Workflows

- A reusable workflow is simply a workflow with a **workflow_call** trigger.
- Like actions, reusable workflows can have **inputs** and **outputs** as well as **side effects**.
- Reusable workflows can be in another repository.
  - The calling workflow must have permission to access the called workflow.
- [Example 3: https://github.com/dvnrrs/github-actions-demo/blob/main/.github/workflows/3-reusable-build.yml]
- [Example 4: https://github.com/dvnrrs/github-actions-demo/blob/main/.github/workflows/4-build-myprogram-with-reusable-workflow.yml]

GitHub Actions - Artifacts

- **Artifacts** are files that can be uploaded by a job.
  - They can later be downloaded by other jobs.
  - They can also be downloaded from the website.
  - They can also be downloaded using the “gh” CLI.
- They are the primary method of sharing *data* between jobs.
- Upload an artifact with the actions/upload-artifact action.
- Download an artifact with the actions/download-artifact action.
- [Example 5: https://github.com/dvnrrs/github-actions-demo/blob/main/.github/workflows/5-automatic-release.yml]

GitHub Actions - Matrices

- A matrix can run multiple parallel instances of a job with varying parameters.
  - The matrix defines the parameter values for each instance.
  - Multiple covarying parameters can be defined.

- [Example 6: https://github.com/dvnrrs/github-actions-demo/blob/main/.github/workflows/6-matrices-and-concurrency.yml]

- [Larger Example: https://github.com/hbkdaq/next-firmware/blob/master/.github/workflows/build-all-impl.yml]

HBK
HOTTINGER BRÜEL & KJÆR

GitHub Actions – Concurrency Groups

- Concurrency groups limit how many instances of a job or workflow can run at the same time.
- Each concurrent job/workflow defines a group string. If two instances’ group strings evaluate to the same string, those instances are concurrent.
- Default behavior is to wait for previous jobs/workflows to finish before starting another.
- With **cancel-in-progress**, previous jobs/workflows can instead be cancelled to let the new one run. This is useful, for example, for pull request triggers – if the pull request is updated, a previously running build should be cancelled.

GitHub Actions – Secrets

- Secrets are protected strings that can be accessed in a workflow via `${{ secrets.NAME }}`.

- Secrets are defined at the repository or organization scope.

- GitHub aggressively strips secrets from logs.

- Secrets can be inherited by invoked reusable workflows.

GitHub Actions – Badges

- Badges are HTTP-fetchable SVG images that show the status of a workflow.

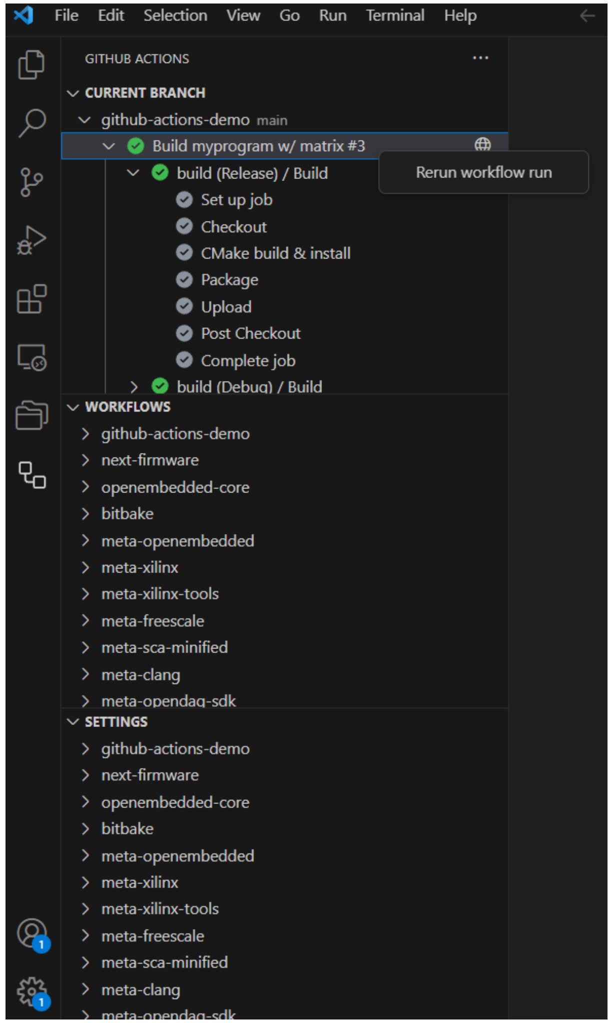
- Construct a URL like:
  - `https://github.com/OWNER/REPOSITORY/actions/workflows/WORKFLOW-FILE/badge.svg`

- Example:
  - `https://github.com/hbkdaq/next-firmware/actions/workflows/nightly-release.yml/badge.svg`

GitHub Actions Demo
passing

GitHub Actions – Visual Studio Code Integration

- Official VS Code extension
- [Live demo]



CURRENT BRANCH
- github-actions-demo main
  - Build myprogram w/ matrix #3
    - build (Release) / Build
      - Set up job
      - Checkout
      - CMake build & install
      - Package
      - Upload
      - Post Checkout
      - Complete job
    - build (Debug) / Build

WORKFLOWS
- github-actions-demo
- next-firmware
- openembedded-core
- bitbake
- meta-openembedded
- meta-xilinx
- meta-xilinx-tools
- meta-freescale
- meta-sca-minified
- meta-clang
- meta-opendaq-sdk

SETTINGS
- github-actions-demo
- next-firmware
- openembedded-core
- bitbake
- meta-openembedded
- meta-xilinx
- meta-xilinx-tools
- meta-freescale
- meta-sca-minified
- meta-clang
- meta-opendaq-sdk

HBK HOTTINGER BRÜEL & KJÆR

GitHub Actions – Questions?

Thanks for your time!