CI/CD with GitHub Actions

Presented By: Parth Patel

Agenda

- What is GitHub Actions?
- Components of GitHub Actions
- Why having GitHub native CI/CD tool is helpful
- Some use cases of GitHub Actions
- Demo

What is GitHub Actions

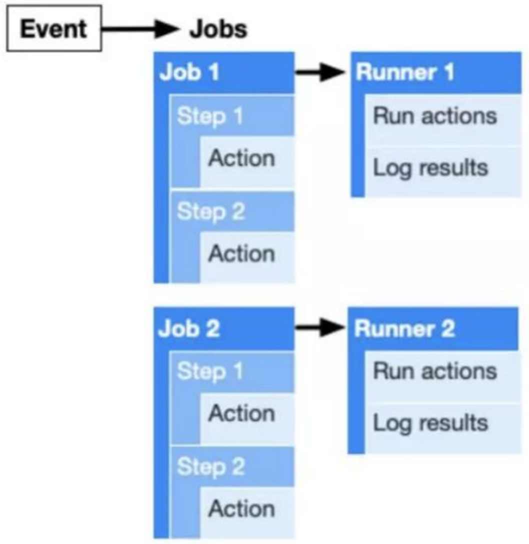
- GitHub Actions gives developers the ability to automate their workflows across issues, pull requests, and more plus it gives native CI/CD functionality.

GitHub Actions

- GitHub Actions allows users on GitHub to perform a large number of actions , such as deploy to AWS, Terraform, email the developers if any commit breaks any tests , and much, much more!
- At the most basic level , GitHub Actions bring automation directly into the software development lifecycle on GitHub via event-driven triggers.
- These triggers are specified events that can range from creating a pull request to pushing any new commit.

Components of GitHub Actions

1. Workflows
2. Events
3. Jobs
4. Steps
5. Actions
6. Runners

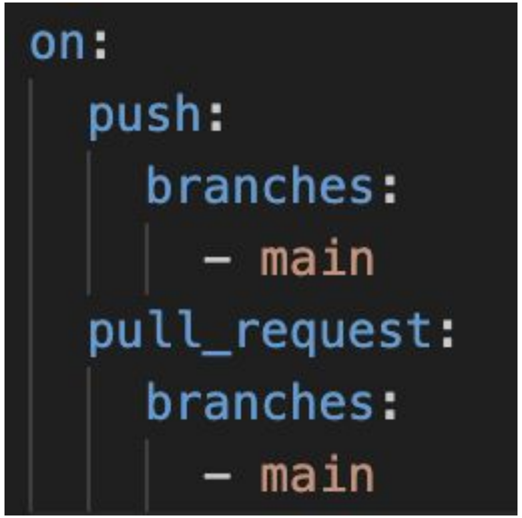


Workflows

- The Workflow is an automated procedure that you add to your repository.
- They are made up of one or more jobs and can be scheduled or triggered by an event.
- It can be used to build, test, package, release, or deploy a project on GitHub.
- Workflows can be created inside the ***.github/workflows*** directory by adding a ***.yml*** workflow file.
  For example, add ***.github/workflows/test-cicd.yaml*** to your project

Events

- An Event is a specific activity that triggers a workflow. For example, activity can originate from GitHub when someone pushes a commit to a repository or when an issue or pull request is created.
- You can also use the repository dispatch webhook to trigger a workflow when an external event occurs.



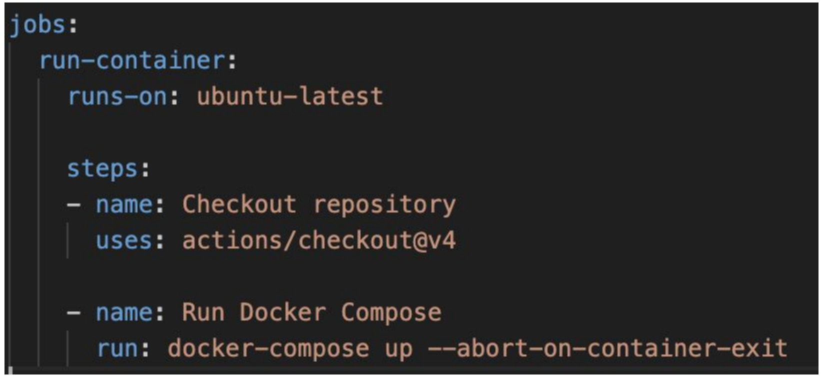
```yaml
on:
  push:
    branches:
      - main
  pull_request:
    branches:
      - main
```

Jobs

- A Job is set of steps that execute on the same runner.
- By default , a workflow with multiple jobs will run those jobs in parallel. You an also configure a workflow to run jobs sequentially.
- For example, a workflow can have two sequential jobs that build and test code, where the test job is dependent on the status of the build job. If he build job fails, the test job will not run.

Steps

- A step is an individual task that can run commands in a job.
- A step can be either action or a shell command.
- Each step in a job executes on the same runner, allowing the actions in that job to share data with each other.



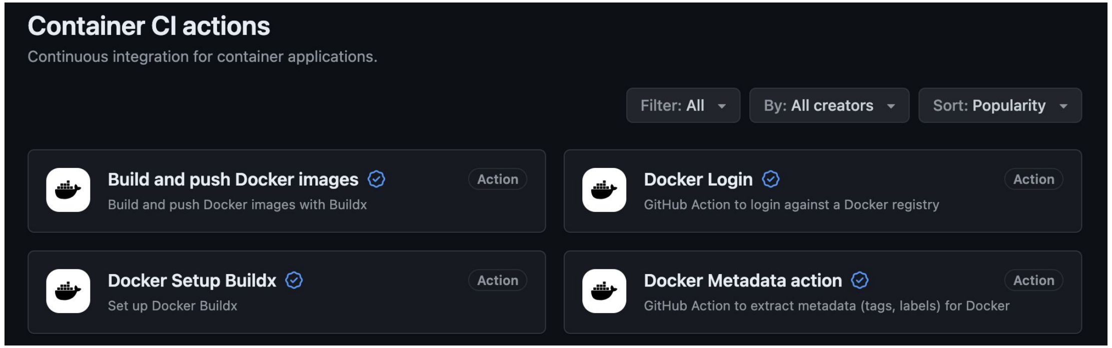
```yaml
jobs:
  run-container:
    runs-on: ubuntu-latest

    steps:
    - name: Checkout repository
      uses: actions/checkout@v4

    - name: Run Docker Compose
      run: docker-compose up --abort-on-container-exit
```

Actions

- Actions are standalone commands that are combined into steps to create a job.
- Actions are smallest portable building block of a workflow.
- You can create your own actions , or use actions created by the GitHub community. To use an action in a workflow, you must include it as a step.



Container CI actions

Continuous integration for container applications.

Filter: All | By: All creators | Sort: Popularity

Build and push Docker images — Action
Build and push Docker images with Buildx

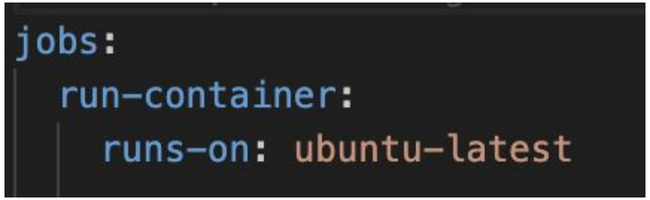
Docker Login — Action
GitHub Action to login against a Docker registry

Docker Setup Buildx — Action
Set up Docker Buildx

Docker Metadata action — Action
GitHub Action to extract metadata (tags, labels) for Docker

Runners

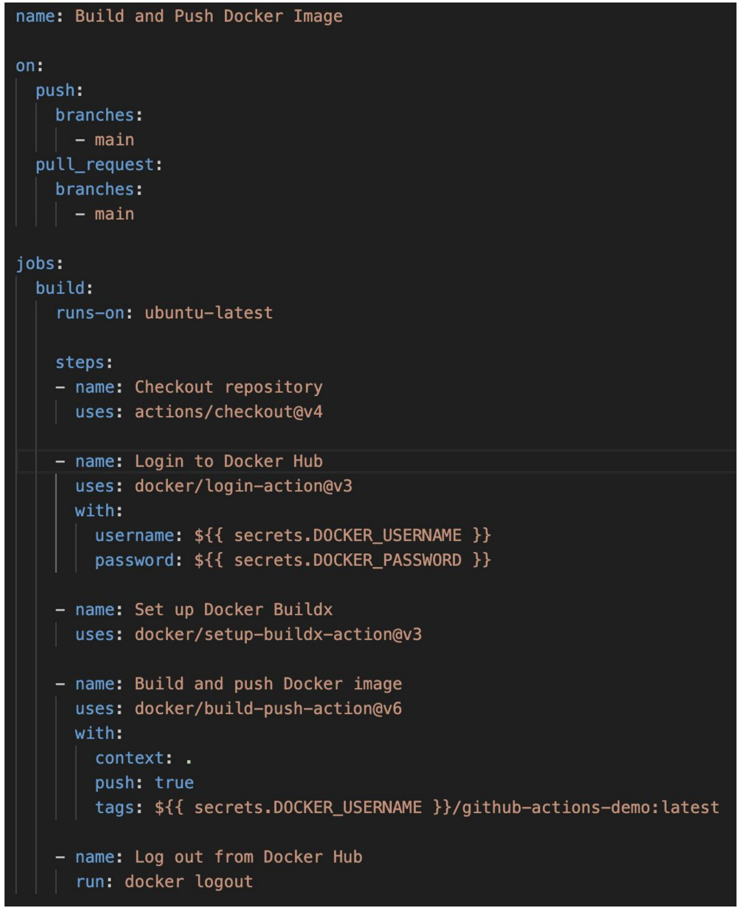
- A runner is a server that has the GitHub Actions runner application installed.
- A runner listens for available jobs, runs one job at a time, and reports the progress, logs, and results back to GitHub.
- You can use a runner hosted by GitHub, or you can host your own.
- GitHub hosted runners are based on Ubuntu Linux, Microsoft Windows, and macOS and each job in workflow runs in a fresh virtual environment.
- If you need a different operating system or require a specific hardware configuration, you can host your runners.



```yaml
jobs:
  run-container:
    runs-on: ubuntu-latest
```

A sample workflow

- Name: Name of Workflow
- On: On what event the workflow should be triggered?
- Runs-on: Where does it run?
- Steps: What gets run?
- Job: Set of steps that execute on the same runner



```yaml
name: Build and Push Docker Image

on:
  push:
    branches:
      - main
  pull_request:
    branches:
      - main

jobs:
  build:
    runs-on: ubuntu-latest

    steps:
    - name: Checkout repository
      uses: actions/checkout@v4

    - name: Login to Docker Hub
      uses: docker/login-action@v3
      with:
        username: ${{ secrets.DOCKER_USERNAME }}
        password: ${{ secrets.DOCKER_PASSWORD }}

    - name: Set up Docker Buildx
      uses: docker/setup-buildx-action@v3

    - name: Build and push Docker image
      uses: docker/build-push-action@v6
      with:
        context: .
        push: true
        tags: ${{ secrets.DOCKER_USERNAME }}/github-actions-demo:latest

    - name: Log out from Docker Hub
      run: docker logout
```

Why having Github native CI/CD tool is helpful

- Build into GitHub
    - GitHub Actions is fully integrated into GitHub and therefore it doesn’t required external site.
- Multi-Container Testing
    - Actions allows you to test multi-container setups by adding support for Docker and docker-compose files to your workflow .
- Multiple CI templates
    - GitHub provides multiple template for all kinds of CI configurations which make it extremely easy to get started.
- Great Free Plan
    - Actions are completely free for every open-source repository and include 2000 free build minutes per month for all your private repositories. If that is not enough for needs you can pick another plan or go the self-hosted route.

Some use cases of GitHub Actions

1. **Build, test and deploy within the GitHub flow**.
2. **Automate repetitive task:** GitHub Actions can be used to automate an endless number of steps in the software development lifecycle.
3. **Easily add preferred tools and services to the project:** GitHub Actions give you the ability to connect and integrate your preferred third-party tools and services directly into your repository.
4. **Keep track of your projects:** You can use GitHub Actions monitor application builds, measure performance, track errors and more via integrations with third-party tools.
5. **Quickly review and test code on GitHub:** GitHub Actions lets you integrate any number of third-party testing tools directly into your workflow in your repository at any step. Moreover, GitHub Actions enables multi-container testing and “matrix builds”, which lets you run multiple tests on Linux, Windows and mac OS at the same time.

Demo

Lets see the things in action…..

Resources

- There is an active community
  - https://github.community/t5/GitHub-Actions/bd-p/actions

- And documentation at
  - https://help.github.com/en/categories/automating-your-workflow-with-github-actions

Thankyou!

Parth Patel

Sr. DevOps Engineer at Crest Data Systems