ENG1 Continuous Integration

Cohort 3 - Group 28

“Team 28”

Muhammed Salahudheen
Joel McBride
Jamie Rogers
Maciek Zaweracz
Rhys Yeaxlee
Alex Spencer
Alex Firth

Methods and Approaches

For our project, we used GitHub Actions for continuous integration as it offers several benefits that makes it an ideal choice for our group. One of the benefits is that, we initially used GitHub for collaboration and GitHub Actions is integrated with GitHub meaning that the integrations and setup processes allows you to manage your continuous integration directly within your repository, making it extremely convenient for our group. GitHub Actions is also very easy to set up as it provides a simple YAML-based schema for defining CI workflows. Also, some of our team members had previous experience with using this platform, which made setup faster and more predictable.

Alternatives include GitLab CI/CD and Jenkins. GitLab pipelines are tethered to GitLab repository history, as GitHub Actions are to GitHub, so that would have made sense if we were using that for our codebase. Jenkins is a separate service that can be integrated with various developer platforms and self-hosted. However, the effort to set up and maintain it means it wouldn’t be worth it since we only have limited workflows which wouldn’t take much time to port if the project ever did need to be moved elsewhere.

In addition to easy setup and integration with GitHub, GitHub Actions provide a large variety of pre-built actions and workflows that can be easily modified and customised to suit our project's needs. GitHub Actions is also free, so it suits our projects cost-effectiveness as we can utilise it without incurring any additional costs. The free services have a limit of runtime minutes, but it’s generous enough that it shouldn’t be an issue for a project of this scope.

Considering the future development of our project, GitHub Actions is beneficial towards adapting and accommodating new features and changes as it is flexible and scalable. Most importantly, GitHub Actions is very popular and it is also very transparent therefore, there is quick and easy aid in identifying issues and resolving them.

Continuous Integration Infrastructure

Since we used GitHub Actions, the CI platform is directly defined within our GitHub repository. We defined our workflows using YAML files in the git repository: .GitHub/workflows. The workflow outlines and checks how it's formatted, runs the tests and validates the build.

These are separated into three different jobs, lint.yml, test.yml, and build.yml. Lint and Test have the same triggers and much the same purpose but have been split since the tests are run on a matrix of different operating systems to ensure stable cross-platform support as specified by our requirements, whereas the linting (Spotless formatter validation) is fully static analysis and can be run once on the most convenient platform.

The CI workflows are triggered automatically on specific events such as pushes, pulls, etc. so that the code changes are continuously and consistently tested and validated. Depending on our project’s requirements, may alter or have additional steps in the CI workflows before and after deployment.

We had notifications setup for activity in the GitHub organisation in our Discord server so that other team members could be notified and stay informed about the status of builds and changes that were made. We noticed there were too many failures occurring throughout the CI builds while setting them up which were congesting the Discord chat, therefore, we set up a different branch to work and add onto the project and only merge to main after major changes so that the changes were more structured and easier to follow. The “main" branch is protected so it can only be added to with pull requests, and we use a separate working branch. Validation runs on pull request contents to make sure they’re ready, and on all commits to the main branch so it’s clear at a glance they’re all valid. The build workflow runs on releases.

Bibliography - sources used for GitHub Actions research

- Lecture slides "Continuous Integration"
- github docs
    - https://docs.github.com/en/actions/automating-builds-and-tests/building-and-testing-java-with-gradle
    - https://docs.github.com/en/actions/using-workflows/events-that-trigger-workflows
    - https://docs.github.com/en/actions/using-jobs/choosing-the-runner-for-a-job
    - https://docs.github.com/en/actions/using-jobs/using-a-matrix-for-your-jobs
    - https://docs.github.com/en/actions/using-workflows/storing-workflow-data-as-artifacts
- actions pages
    - https://github.com/actions/setup-java?tab=readme-ov-file#usage
    - https://github.com/gradle/actions/blob/main/setup-gradle/README.md
    - https://github.com/actions/upload-artifact?tab=readme-ov-file#usage
    - https://github.com/softprops/action-gh-release?tab=readme-ov-file#-usage
- local testing
    - https://nektosact.com/usage/index.html