Continuous Integration Report

Group 28

Piazza Panic

By OuseWorks (from devCharles, Group 26)

Ben Harris
Joshua Gill
Niamh Hanratty
Amy Raymond
Matthew Czyzewski
Matt Rohatynskyj

Approach to CI

A single source repository was maintained on GitHub which allowed CI concepts to be applied to the project's development. The source code for the game and the website were maintained in two separate GitHub repositories that were forked from the previous group.

Github was chosen over several other options such as GitLab and Bitbucket. The platform was found to be very appropriate for this project because of the convenient tools baked into it called GitHub actions, pages and projects which aided the project's development and organisation.

Throughout the project team members integrated their work frequently to the main branch of the repository so that everyone was kept up to date with the progress, problems were smaller to fix and did not have a chance to build up over time.

GitHub Actions - Java CI with Gradle Workflow

A GitHub Actions workflow containing build and release jobs was created in the game repository. Workflow artefacts were generated and are viewable on GitHub.

Build job for pull requests

During development, pull requests were made before code was integrated into the main branch. This allowed changes to be checked for errors by an automated build before integrating.

The team configured the build job to automatically run when a pull request was made or a branch was merged into main. The build job carries out the following actions:

- Check that the project builds in the latest Ubuntu (Linux), MacOS and Windows environments.
- Run unit tests then generate a viewable testing report artefact using the Jacoco Gradle plugin.
- Generate executable jar file artefacts in Ubuntu (Linux), MacOS and Windows environments to check that it is supported by multiple operating systems.
- Generate a Google checkstyle report artefact using the Gradle checkstyle plugin to inform the team about how to make the code more readable.

Release job for new versions of the game

The release workflow was designed to run after the build job if a git tag specifying a new version was present. The release job would download the executable jar file artefacts for each OS environment and then include them in the release.

GitHub Actions - Website Workflow

A workflow was also created to automatically build and then deploy the website.

Continuous Integration (CI) Infrastructure

A brief report of the CI infrastructure that was developed for this project is outlined below to demonstrate how the team's approach was applied in practice.

Each of the images referenced in this document are viewable on the CI section of the website. The CI infrastructure is also viewable by visiting the game's GitHub repository using this link. The full gradle.yml workflow file is shown in Figure 2.

- **Figure 1** is an example screenshot from GitHub showing the output from the execution of the build job in the Java CI with Gradle workflow.
    - Downloadable artefacts for the executable jars, Jacoco testing report and checkstyle testing report are visible in the image.
    - The image also shows evidence of the build job being carried out successfully in multiple os environments.
    - In this image the release job was not triggered because there was no git tag indicating a new release present.

- **Figure 2** presents the game's full gradle.yml workflow as a screenshot of the file with comments included to explain all the key steps.

- **Figure 3** is an example of a release of the game made automatically by the CI workflow release job after the **v0.0.1** tag was created.
    - The release can be seen to contain the executable jar artefacts of the game built in the different os environments. The source code is also available as an artefact.

- **Figure 4** demonstrates the sequential execution steps of the build job workflow in action.