Continuous Integration Report

Cohort 3 Group 6 - Carbon Goose

Members: Bailey Horsley, Owen Jones, Rory Ingram, Ken Jacob, Abishek Kingslin Raj, Louis Polwarth, Adam Johnson

Continuous Integration Report

Continuous integration approaches:

- First of all we make sure to keep in frequent contact with each other when we are developing the project meaning that we are able to develop different parts of the game simultaneously effectively.
- We used a single repository for the creation of our game, which is located on GitHub allowing for easy access to our code for development and allowing changes to be made and shared easily allowing for us to easily work collaboratively.
- Along with this we try to push our changes that we have made independently on branches frequently to the main branch ensuring that we are all working on the newest version, resulting in minimal integration issues when merging new features together.
- Whenever we had a build that did not pass the automatic tests, we aimed to have this build be fixed as quickly as possible.
- Whenever a change was made to the GitHub repository, we received notifications on our Discord server, as well as notifications reporting the result of the Actions Pipeline, i.e. builds failing or succeeding.

Our use of GitHub Actions:

- GitHub Actions to make it so that each time we push or merge to the main branch an automated build will be performed, on Linux, Mac and Windows meaning that we will quickly know if our merges have caused any build errors and ensuring that we know our game works on the three major operating systems.
- After this build has been created the automated tests will be run on each operating system, checking the reliability of the build, and a test report will be generated as a product of this quickly showing us where any issues with our most recent build are.
- Each time we pushed to main we could see the results of if the build was successful and the automated tests were successful in the actions tab of GitHub allowing us to quickly tell if this new commit was successful.
- Another feature we have on our GitHub Actions is that if a “build” tag is pushed after a successful build and the automated tests pass a release will be created with the generated jar files. Allowing us to be able to easily obtain a jar file for our game which allowed us to let users try our game and give feedback.
- These GitHub Actions helped us be able to effectively and quickly find any issues in newer versions of the game that we created. We found it very useful especially when performing a pull request to main, as this would automatically run the tests before merging, meaning we could find issues with our branches before merging to main. Allowing us to diagnose issues before being put on the main branch not inconveniencing others trying to use the main branch of the repository.
- Another important part of these GitHub Actions was to ensure that the time for this to take place was quick, as we wanted to know if our new build was successful. This was completed by making the tests on the three operating systems being performed in parallel. This resulted in each time these GitHub Actions were performed the response was much quicker.