SENG201 Project Report

Authors: Lorenzo Fasano (Student ID:) and Jay Hamilton (Student ID: 84258116).

This report highlights the structure, testing, strengths and weaknesses of the application HeroesAndVillains.

The focus of this project has been to implement the basic design principals of object orientated programming in Java, to familiarise ourselves with the Java API and to develop ways of collaborating on a small scale software project.

The project contains 81 Java classes and 41 JUnit testing classes creating flexible, modular and maintainable code. The main classes present in this project are Character.java, Collectable.java, Minigame.Java and Building.java, these four abstract classes are the backbone of the project’s structure and characterise the expected behaviour in the child classes. Most of the remaining classes such as Hero, Villain, Hospital, RockPaperScissors, HealingItem, and Armor extend one of the four initial classes when constructed.

In the core structure of the project the use of enumerations was fundamental to storing and distinguishing between objects of the classes that implement Character.java, Building.java or Collectable.java. In order to implement the ability to store objects in the Shop’s inventory or in the HeroesSquad’s backPack , the class Inventory was implemented, the main property of inventory is a HashMap<Collectable, Integer> that stores the type of Collectable item and the quantity of it. Supporting methods were then created to check if a collectable was in the inventory, to return a list of the inventory objects and quantities or to add and delete elements from the inventory.

A similar concept was applied for the creation of the HealingWard class, used inside the Hospital to store the information of those Hero objects that are healed using HealingItem objects. In HealingWard the main property is healingWard a HashMap<Hero, Integer> which stores the hero being healed and the time it will take until that hero is fully healed. The time updates in the HealingWard property is achieved using a secondary thread and constantly decreasing the Integer stored for each Hero object.

The communication between the HeroesSquad object and the Building objects was possible by creating in each building a series of methods that used the HeroesSquad object setters and getters and modify this object depending on events. At a higher level the existence of the Engine class and helpers allowed to keep track of the HeroesSquad object, the right City and the Villain of each City, this was done using GameWindowManager in the GUI application as explained in the next paragraph.

Once the command line version of the game was created the GUI version was started. The class GameWindowManager deals with opening and giving visibility to the remaining windows in the right order and it is the responsible handling the HeroesSquad, Villains and list of City objects (world) data among the windows.

The GameWindowManager class also contains the main logic behind the implementation of the serialisation of the game current status and the scores saving.

Junit5 has been used for testing, Junit extensions @BeforeEach, @AfterEach and @RepeatedTest(integer) where particularly useful for speeding up the tests creation and to make sure that tests were run independently from each other. Only after thoroughly testing the low level classes the higher level functionality, such as the creation of a City or of the HeroesSquad objects, was created and then tested. The method interact() present in objects that extend Buiding.java was useful for the creation of the command line version of the game and, once the game was finished it remained important to be able to test the functionality of each Building, HealingWard, Inventory, HeroesSquad, City and Villains objects. The test coverage ended up being SOMETHING%, the modularity of the code helped having high testable code, however, no Swing window has been tested, in the next project more attention will be put in the discovery of new techniques to test the GUI components.

The overall result of this assignment has been both satisfying and rewarding as it was a significantly bigger project compared to what we have done in the past, the collaboration has been intense and a significant effort has provided both partners. Agreed deadlines were respected and the communication consistent, the use of GitHub as a version control platform facilitated easy code updates and backups.

This project has been incredibly time consuming as both parties were new to software development using Java, scaling techniques and testing, this project helped both parts to learn how to use API documentation and how to seek help whenever things did not work properly, this is probably the most valuable part of this experience.

Both partners agree that they contributed 50% each in the creation of this project, this includes testing, Javadoc, UML and general procedures such as the game design and the architecture implementation.

Lorenzo Fasano: My focus has been the creation of city, collectables and part of the engine packages and related Swing windows, testing and Javadoc. The most laborious processes were the creation of reliable low-level classes and the communication between the HeroesSquad object with each building.

Jay Hamilton: My focus has been the creation of characters, minigame and part of engine packages and related Swing windows, testing and Javadoc. The hardest part of this part of the project was to create the minigames functionality and integrate it in the game.

The project is on GitHub as a private repository, anybody reading this document is invited to request access to this repository to have proof of the what has been stated and to see the effort put into this highly rewarding project.

A total of over 400 hours has been spent on the realisation of the project and both students were satisfied with the difficulty of the project and with the learning curve over this period.