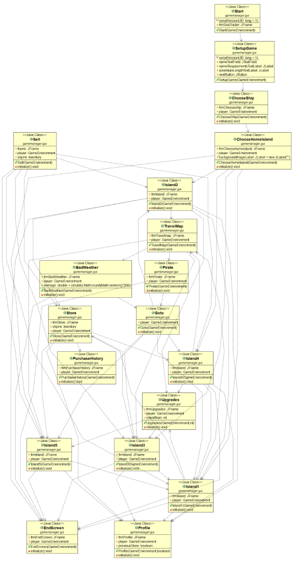
SENG201 Sea Trader Project

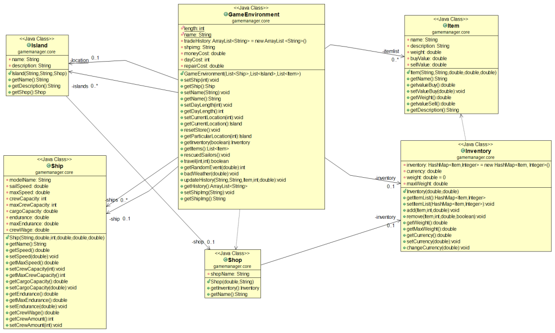
Team 21

Members:

Yaoce Yang 81137152
Harper Liu 83536521

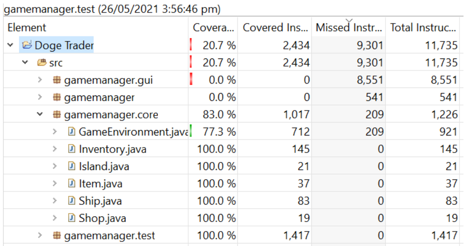
Application Structure:





We have generated 2 separate UML Class Diagrams. One (left side) contains only GUI classes to show the relationship and flow of the GUI windows and the other (above) shows the relationship and flow of the core classes of our program. All GUI classes have a relationship with the GameManager class within the core. The whole UML diagram with both core and GUI classes made the UML diagram unreadable hence we decided to split them apart. Our program functions by firstly creating one GameEnvironment class which holds all of the current game lifetimes information. This is then passed throughout the many GUI classes where they are able to access and set target variables and structures as they please through get, set, and the various other methods.

Testing:



gamemanager.test (26/05/2021 3:56:46 pm)

Element	Covera...	Covered Ins...	Missed Instr...	Total Instruc...
Doge Trader	20.7 %	2,434	9,301	11,735
src	20.7 %	2,434	9,301	11,735
gamemanager.gui	0.0 %	0	8,551	8,551
gamemanager	0.0 %	0	541	541
gamemanager.core	83.0 %	1,017	209	1,226
GameEnvironment.java	77.3 %	712	209	921
Inventory.java	100.0 %	145	0	145
Island.java	100.0 %	21	0	21
Item.java	100.0 %	37	0	37
Ship.java	100.0 %	83	0	83
Shop.java	100.0 %	19	0	19
gamemanager.test	100.0 %	1,417	0	1,417

The image on the left is the screenshot of our application test coverage report regarding our unit test classes. The total coverage across our application is **20.7%.** Our coverage is low due to the inability to test methods that incorporate the GUI window elements. This was because we could not trigger events to happen with JUnit test methods. Hence, we completed a large amount of user testing to ensure that a wide variety of sequences and inputs will not break our code.

Project Reflection:

At the beginning we underestimated the difficulty of the project in all aspects but especially design and implementation. We found these two areas the most difficult as we had used a lot of time at the beginning designing our UML class diagrams which had to eventually be revised as the structure we wanted did not operate well with the GUI. This was primarily due to our lack of knowledge on the WindowBuilder tool as well as the lack of resources available to teach us how to develop our GUI windows. We also wanted to develop our GUI along with our core classes but soon found out it was better to create the core classes first and then move on to GUI development. One aspect of the project we wanted to but could not complete was implement a CMD line UI before our GUI. But half-way through its development, we had to scrap the UI part due to poorly estimated task time distribution and immediately start working on our GUI.

One feature we regret not implementing from the start was to have one window frame throughout the whole lifetime of the GUI window. We realised too late into our project which could have made our window transitions more fluent than our current transitions which dispose and create a new window. However, we are proud of our efficiency regarding task distribution between team members as we believe we both invested the same amount of effort and time into the project while maximising our own strengths and weaknesses regarding the task allocation. We also both enjoyed the process of game and GUI development and learning about the Java coding language. Next time, we want to explore different GUI development methods not using the WindowBuilder tool. This is because we felt WindowBuilder was out-dated and believe there are better and more refined tools available that can provide a better UX.

Effort Distribution & Contribution:

The first 2 weeks of the project, we met up constantly to discuss the game and project requirements and design as well as create the UML diagrams. We spent approx. **15 hours** each on the requirements and initial system plan diagrams. The following 4 weeks we started the construction of the core classes as well as the CMD line UI. We spent approx. **30 hours** each creating the core classes and UI (which we eventually scrapped). The final 3 weeks of the project, we created the GUI and completed unit and user testing on our classes. We spent approx. **70 hours** each completing these and fixing bugs up until the submission deadline.

In total, **each member** was estimated to spend around about **115 hours** total on the project.

We have both mutually agreed that both members have contributed evenly to the project being **Yaoce 50%** and **Harper 50%**.