2021

Island Trader Game

SENG201 - PROJECT

MOHADESA

zkh22@uclive.ac.nz | msh233@uclive.ac.nz

25/05/2021

About this Project:

This is an assignment from University of Canterbury, It's a strategy game in which user/player plays the role of the trader, they can travel by ship to trade good on different islands with the aim of earning as much profit as possible before the game ends.

Design and Architecture:

Our approach to the project was more like agile process, we tried to develop very basic models and methods, test them and then move forward add more functionality and then test again. We made sure that user can perform all the basic actions like make a purchase, sell an item, view all the islands etc. before we moved to the creation and calculation or routes and events.

- **UML and Use-Cases:**
  We started by creating a very basic UML design and use-cases for independent models like, Trader, Ship, Island, Store, Route, Item, Event, and Game Manager. Then as we implemented command line module, we made changes as all the UML classes as needed.

  The use-cases diagrams were bit different, they didn't need as much altering as UML, but we added more use-cases as the functionality of our project/game increased. Please see the UML and Use-Case diagrams for more details.

- **Interfaces:**
  Command line and Graphical instance use Manager-Interface since they are two different implementation of Trader Island game. There are many common methods between command line and graphical game, so it made sense that they both implement one interface.

  Design and logic are kept separately in graphical part of the game. Inputs are passed to game manager to be validated and processed so that the screens can view the result.

  Routes and Events:

  Once we have completed the basic operations, we started to work on providing the "player" with all the possible routes from current island to the chosen destination. We used back-tracking method learnt in COSC-262 to produce all the possible routes that user can choose from. Then we started working on all the possible events, we created 4 sub-events Pirate, Shark, Bad Weather and rescue-sailors which extends Event. We decided to setup the events in 3 different ways:

  - If "player" chooses direct route:
    They will face two events, 1st will be pirate attack and if they survive, then 2nd will be randomly selected.
  - If the route is greater than the direct route and shorter than longest possible route:

The “player” will face exactly 1 randomly chosen event.

- Otherwise:

  The “player” will face at most 1 randomly chosen event other than pirate attack.

- **Graphic User Interface:**

  We decided to keep the GUI very simple and make sure it works and perform all the required functionalities. We created an abstract class Screen and multiple sub-screens extend that class. We spend lots of time on updating the frame/panels but it kept crashing so gave up on that idea and followed on with going to new frame after user makes a choice.

- **Unit Tests:**

  The J-Unit test cover about 50% of the whole game. Which is mostly for the individual models, the problem with the rest of the code is that for command line almost every method/function is printing an output so if we add a J-unit test for those methods we get an output on the console as well and writing a new test method for every function is hard as due to the dependencies of functions/methods and hard to control the hierarchical flow, we had similar issues with Gui.

Feedback:

Overall, the project went well, there was lots of learnings and finding our strengths and weaknesses. As my project partner doing 5 subjects and myself busy with other assignments and baby at home, we always struggled with managing time as we both had other assignments that we needed to focus on as well.

What we learnt in the 2nd term would have been very helpful before we started the project, as by that time we already had completed 50 percent of our project and that project management and planning parts could help us maybe plan and execute better.

We spent 15-25 hours per week on this project during the break, after the break it was roughly around one day per week to work on GUI. Both of us are happy with the amount of work we’ve done individually and number of hours we’ve put in. Lastly, we feel like, the 25% allocation of marks is not enough for work that needed for this assignment.