Advanced Experiment Design in Unity

Jan Kettler, studentnumber: 979374

Link to the video: https://myshare.uni-osnabrueck.de/smart-link/588f8512-99ed-4a93-9b54-c845131f1839/

Introduction

Two friends of mine (who are not enrolled int these courses) wanted to develop a game, and asked me to join them in programming. Since the plan was to program it using unity, I joined them. The idea was to split the workload so that one would be mainly concerned with graphical design, the other one with story, gameplay and a bit of coding while I would mostly code.
During the semester we developed a basic prototype mostly consisting of a tilemap, character and movement control. But soon we noticed that we were loosing the overview over our code, since we had different opinions and ideas on how to structure our code and tackle problems we encountered. These conflicting ideas led to us both not quite knowing what our code was doing at times. So we decided to pretty much start anew with a bit more experience and a better idea of how the game should look like. So we created some UML diagrams depicting how our coding process should look like.
Due to time-constraints and the deadline of this project I started coding on my own, carrying over some of the ideas we already had and orienting myself at the UML. Still ultimately this second version of the prototype is 90% coded by myself. The Unity version used is Unity 2019.3.12f1.

The Game

The game, InnerDemon, we are working on will be a sort of top-down strategy game, with role-play elements, character development and the ability to build your own party. Overall it is heavily inspired by the Pen&Paper style and supposed to give the player a lot of freedom and the ability to make their own choices which influence their experience of the game.
As this is quite a big Project we intent to keep developing and hopefully release it at some point in the future.

Work process and difficulties

Of course there were the general difficulties of, getting to know unity and its editor and all the different functionalities it already brings with it. Secondarily using a new programming language can always be hard at first. Surprisingly I did not encounter a lot of problems with c# as I think it is quite intuitive to use. But I often found myself not completely understanding some unity functionality or just not knowing about it, which in turn led to making code unnecessarily complex or inefficient. As I think that is to be expected though, I wont concretely talk about that in the following.

I started out by making a sample Tilemap with support for different tile types, which behave slightly different (as a basic world for testing functionalities).
I imported a player and started on object selection functionality which is among others used to find a path from our player to the selected object. For that I use an A* algorithm that returns the shortest route, if one is possible, depending on the properties of the tiles traversed. The selection is combined with automatic highlighting of the path to give an indication how it looks. This part especially bothered me over and over and a lot of fine-tuning had to be done to make it work properly. The biggest issue is that the highlight has to be correctly removed at certain occasions to prevent the whole map from getting highlighted.
Next came the movement control for the units. I tried many different strategies to make the unit move smoothly along its path and ultimately stuck with using coroutines, since using update seemed difficult.
I added a enemy unit with basic behavior and later an additional playerUnit script both of which

inherit from the basic unit.
Then I started working on a basic UI, displaying a combat log and Characterportrait. The combat log is a scrollview that writes messages and informs the player of their actions and what happens in their surroundings. The Characterportrait displays a closeup camera of the corresponding unit and a slider component, displaying a healthbar with the units current health. There is also a Characterportrait showing enemy units.
I struggled a bit with the UI because I was unfimiliar with the structure and a lot of fine-tuning is needed to get it approximately right. At that point I grew a bit frustrated with all the different references needed and decided to include a Game manager and UI manager singleton to keep important references between objects and use them as a port to interact with different components.
The UI manager keeps references for the UI and handles all calls to interact with the UI.
The Game manager stores references to the map and units and handles interaction between them. Also it handles some game logic and stores the turn queue determining the sequence in which enemy or player units get to act.
I also implemented a basic moba like camera that allows to scroll when the mouse is close to the screen borders.

Generally I attempted to lay the foundation for our game, focusing on core mechanics and reusability of code, since there is much more content to come. So the project is by no means finished now and not really supposed to be playable, which I hope is understandable.

Evaluation

The course was easy to follow and insightful. I was able to get a good understanding of unity and the programming language C#. I especially enjoyed the opportunity to work creatively and pick a subject I am interested in. I learned that when working on a big project some things are much more important than others, for example comprehensible code.