F21GC - Games Production and Playtesting Evaluation

By Jorge Torres H00362596

Introduction

Even though my strengths are not in the production aspect of game development, I still understand the great importance of prototyping, playtesting and evaluating coding techniques in the game development cycle.

I have the facility to understand complex technical concepts in programming, artificial intelligence and security due to my background in computer science though my ability to implement these concepts in a correct way isn’t my forte.

In these prototypes, I will show how I have used an extremely simple prototype in order to evaluate and explore the behaviours of crowds in a navmesh as well as producing a final state machine that is efficient.

Requirement Gathering

For the first prototype, the requirements are to create a crowd, with different groups within that crowd and analyse the behaviour they have when the navmesh is changed in the level. This is in order to evaluate the time they take to change their path if they do at all.

The second prototype is more complex. Having animations and effects my aim was to link the animation tree to the intended behaviour and possibly add an extra type of behaviour.

I had to read through different libraries in order to obtain all the syntax examples I needed to complete the final state machine. I decided to do a singleton in order to keep the code clean and efficient.

Methodology

The way I approached the FSM was with events. These being Enter Update and Exit. Each behaviour had different conditions that had to be made to jump from event to event and those conditions are the ones that define the NPC’s behaviour.

Using a singleton was very useful as not having to get the positions of each point of interest every time the behaviour was triggered made the application run as intended rather than be laggy.

It was my main objective to develop a prototype that demonstrated my knowledge of efficient programming, and the comments I make on the scripts describe what things I could have changed to make it run smoother, but due to lack of time and resources they weren’t implemented.

I am able to code when necessary though I prefer not to as I believe my efforts are more valued elsewhere in the game development process. Nonetheless, this is an example that I can code when needed and that I can easily learn new concepts and find reading code very easy and I am able to quickly recognise what the programmer’s intentions were.

There are many ways of implementing finite state machines such as using a singleton (the method I chose) using a stack-based method and others. For this prototype I focused on the singleton method due to its simplicity and that it was a great way of me being able to demonstrate my ability in coding.

I always have the user in mind and apart from having a well designed and thought game, it is crucial that the technical aspect of the game goes hand in hand with the design view in order to deliver an overall consistent experience.

I was troubled in the last section of the FSM where I intended to add one last loop to the already existing behaviours. This was in order to trigger the sleeping animation once the NPC had approached the safe zone. Even though I have coded the event for it to do so, it failed to stay in the safe zone for long at all therefore not being able to play the animation.

With more time and resources I am sure this would have been resolved.

Risk Mitigation

Risk management and recovery is a massively important aspect of video game production. Games are such complex pieces of software that not having a proper risk mitigation strategy can result in losing the work of dozens of developers, artists and designers and have nothing to show for these efforts.

In this case, being such a small project I did not have to utilise risk mitigating tools such as GIT or others, but I still faced risks and had to recover from them. The problem I faced was due to my computer shutting itself down and one drive updating, it bugged my file system and therefore unity was bugged. All structures in the scene view were bright purple and the unity crashed.

Rebooting the project into safe mode was not a solution as it was still unusable, and remained so until I managed to move the scripts and redo the scene. It was a frustrating experience but I managed to resolve it in the end.

It is extremely important that we always have an up to date, safe back up copy in case any software, hardware physical or mental emergency results in the developers losing much of their game.

Issues

I have faced many issues in the last two months. Not having a working computer as well as dealing with the problems everyone is facing in these covid times has proven to be tough but I feel like I have done well to come out the other side and still produce quality work.

Combining a masters degree and the life of a semi-professional football player has also proven to be more difficult than I initially thought. Struggling to make room to make proper meals, train and take care of my body, recover from injuries and deal with transfers; all while doing my work for university has made me reconsider what I am capable of and make me feel proud of myself for how far I have gone even in this global pandemic.