Reflective Report

My roles and responsibilities during the SE phases

Our team decided to create a 3D tower-defense game using Unity. Although some our team members had experience with game development, none of us have worked on a 3D game that has that many elements involved in it. I have an interest in artificial intelligence, so at the beginning of the project I immediately suggested that I could be working on enemy AI – taking the given path, changing the states between walking and attacking the player and so on. At the first week or so, we all tried to approximate how much can be done in two months, considering that it is not a lot of time for developing a game and there is only five of us, not even professionals, but students working on it. That was super important for us, because we did not want to get too ambitious trying to add too many elements to the game and failing miserably – submitting a game having loads of half-working functionalities. Our idea was clear – creating a basic version of the game and adding extra features if we realise we still have some time left. The reason I am telling this is that I soon realised that for our basic version of the game there would not be much of an AI, so I suggested that I could also take the role of animating characters in the game. I read some material and watched tutorials on character animation and thought that I would not be able to create all the animations in two months by myself, but still, it was my responsibility to implement animations before the deadline. After some research I found an Adobe platform called Mixamo, which has a variety of characters and animations to use and it is completely free. I downloaded some movement animations and the basic character, because I knew it will take some time for my teammate to design our main character. The flexible implementation was necessary to make the transition between downloaded character and newly designed character as easy as possible. It was also my responsibility to make animations run smoothly and have transitions between each of them without weird looking snapping. My team member provided me the character movement script, but I had to modify it to make it work in flow with the animations (e.g. adding the appropriate velocity on y axis to make the character experience realistic gravity along with the timing of the jumping animation). Another responsibility I talked earlier about was enemy AI. I would not necessarily call it artificial intelligence in this particular case, but still, I had to make enemies work on their own. Our plan was to design several levels, thus I could not hardcode coordinates for the enemy, because different levels would have different paths depending on the level’s layout and this solution would either cause glitches or require several scripts for the same thing, which is not a really good software engineering practice. Therefore, my responsibility was to implement the pathing system as flexible as possible. Besides that, I had to make enemies interact with environment. For example, when the player gets close enough to the enemy, it starts attacking the player and when the player runs away, the enemy continues on its path. When my teammate finished main character and weapon designs, I had to rearrange animations and animation controllers to make it run smoothly and I was also responsible for making the gaming experience more enjoyable by adding shooting effects. Our team did not have one member responsible for testing, so each of us had to test our own functionality and later test combined pieces of software from each member. I printed the coordinates of various points in 3D space making sure that my vector calculations were correct in the script, I also used Ad hoc testing technique, which is basically trying to break the system by random error guessing (i.e. simply playing a game searching for flaws) as well as applied my scripts to different objects and changed their parameters to see how the implemented functionality acts with various parameters. And some other techniques, like regression testing (i.e. re-running tests to check if recently added functionality did not introduce new vulnerabilities) were used to make sure it would still work when my teammates add more components to the game.

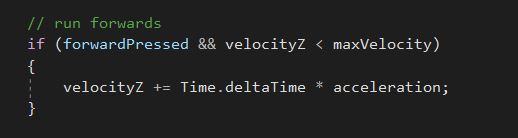
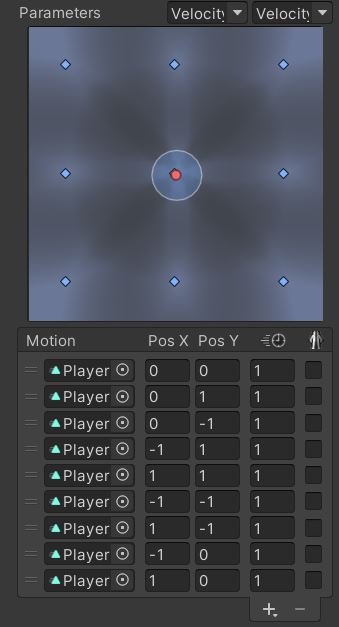
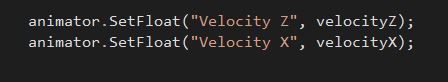
Skills developed and improved during the project

I am really grateful for the opportunity to work on a group project for two months, without having other modules, allowing myself to put 100% of my focus into this project. I remember one of the module leaders saying in the meeting that we should treat this project as a full-time job and I did not put enough emphasis on these words until a month later I told my friend that I actually feel as though I am a full-time game developer. It is not necessarily a skill I have developed, but I think this realisation is as important – it allows me to make better decisions on which path of computer science do I want to pursue after the graduation. During the project, I developed and improved a lot of skills. The fact that we were creating a game meant that each component of the application was very dependent on each other, thus we had to establish good communication between team members. I had to trust my teammates and they had to trust me. If one of us had difficulties, others were eager to help. This boosted my teamwork skills to another level compared with all my previous projects. My roles involved quite a lot of scripting, therefore I improved my programming in C Sharp skills significantly. To achieve my tasks, I had to include numerous mathematical calculations (e.g. doing operations with three-dimensional vectors, calculating angles and distances) in my scripts, thus I strengthened my mathematical knowledge, especially linear algebra. I gained a lot of experience of using very popular, sophisticated game development engine Unity and its various tools and packages. I spent most of the time working on animations, so I developed skills and understanding of how to import animations, how to control and modify them and how to actually apply them in real game. I believe me being a good software engineer is measured not by how much do I know about this topic, but how fast can I come up with a solution to the problem and what quality is it going to be. So, two more skills that I improve doing almost any software engineering project are information finding and problem solving. This project improved those skills to a higher degree. I spent countless hours searching for information, watching tutorials, reading the documentations of various functions and classes, it helped me to understand how to navigate through particular tools, how to properly use those functions, what are the differences between calculations in two-dimensional and three-dimensional spaces and so on. I was stuck on some problems for days, tried several ways to approach the problem and even if it did not work, I still knew I had to solve it and eventually I always found a way to work around it. Putting that much effort not only enhanced my problem-solving skills dramatically, but also increased my patience.

Problems encountered and how did I attempt to solve those problems

I have encountered a huge amount of problems throughout the development and a lot of them required different approaches. Describing all of them would make it too boring to read, but here is some of them:

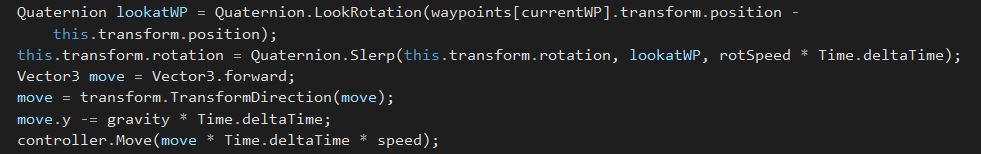
When I implemented character movement animations I realised that the character snaps between different direction animations. When I pressed W and A keys back and forth, the character would snap between running forward and running left without smooth transition. My solution: I implemented two-dimensional blend tree, which controls how much impact does each direction has on each animation and created two variables velocityX and velocityZ, so that the keys continuously increase velocity in that direction till it reaches the limit (i.e. the animation is running on full speed). This made the animations work continuously.



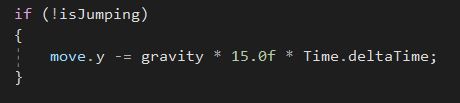
Script running forward

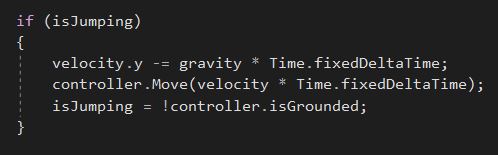
Passing velocities to Unity’s animator component

Blend Tree (with 9 different animations)

To make enemy pathing flexible I used waypoints system. Our game’s paths are not just straight lines, they have different angles, so enemies used to reach one waypoint and snap to the direction of upcoming waypoint. My solution: I used a class called Quaternion, which allows the object to transition from its current rotation to the given rotation with a certain speed, instead of instantly.

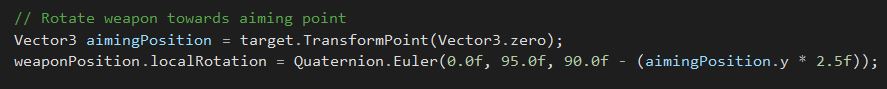
I implemented gravity, while working on character jump. When I used jump, the character fell to the ground at the reasonable velocity. Everything looked fine until I moved around the map. When the player walked down the stairs or simply walk off the fence its falling velocity looked slower than it should have been. My solution: I basically implemented different velocities on y axis, depending on the state which the character is in. One state - character is jumping, another state - character is falling, because it has just stepped off another object.





My teammate designed the main character, I imported it into our Unity’s project and when I tried to implement animations, they looked really weird with bones being deformed. After some research I realised that character rigs, which are used for animations, have different types. And my previous, downloaded character had a generic rig type, which caused glitches to the newly imported character. My solution: I changed the rig type of the previous character to humanoid, then converted all the animations from generic to humanoid type using previous character, then applied humanoid rig to newly designed character and then inserted these new animations into our designed character’s animation controller.

We wanted to make our character’s upper body bend as it points the weapon upwards or downwards, but while implementing that, I came up with loads of other glitches, so I came up with an idea to rotate the weapon instead of its upper body. I encountered two main problems while doing that. The first one was that the weapon could only rotate around its centre, which did not look good and the second one – how do I calculate the point where the player is aiming. My solution: I solved the first problem by creating an empty object, placing it at the upper back corner of the weapon and then parenting it to the weapon object, so when you rotate the recently created object it rotates the weapon around the pivot at the end of it. For the second problem I created an empty object and made it a child of the camera object (our camera moves according to mouse), aligned it in the middle of the screen (i.e. where the player is aiming) then in the script I transform that empty object’s coordinates into world space and use its y axis position to change the rotation of the weapon.



How has this module improved my employability skills and portfolio of work

I have already mentioned some skills I developed during this project that contribute to being a good employer. Software engineering projects are rarely being done by a single person, thus communication skills, that I improved during this project, is a necessity. Some problems I encountered had my mind occupied for days, which helped me to increase my patience and I believe patience is an admirable quality of a person, because it helps while communicating with others as well as solving hard problems. Almost every part of the project was closely related to one another, therefore I tried my best at being trustworthy and reliable to my teammates. Usually I am not the type of person, who takes initiative to lead the group of members and push them to do their tasks on time, but sometimes it is duty. Even though I was not our team leader, who did a phenomenal job at it, I motivated and helped my other teammates, suggested ideas and solutions, encouraged them to finish their tasks as soon as possible, because for me it is more important to deliver a high-quality software application as a team than to deliver a high-quality piece of software that I was responsible for. It is not like I motivate people and do not do my own tasks properly, but I just never work by the attitude “I did my part, I am happy with it and I do not really care about yours, unless it affects the evaluation of me”. I truly believe this kind of attitude was a big factor in us achieving what we have achieved in only two months and it really boosted my leadership abilities. Besides that, I was also very determined to follow the instructions of our team leader and if something was unclear I just asked, because I think it is better to ask, if you do not know, rather than start developing with uncertainty and confusion. I was questioning ideas, if I felt that a different approach was more appropriate, we discussed their advantages and disadvantages and eventually agreed on which path do we want to take. The project improved my ability to listen, I was open to any suggestions directed towards tasks, that I was responsible for, considered them and applied the ones that I thought would benefit our game, which also strengthened my decision-making skills. One of my strengths is adaptability. My attitude is “what programming language is the best for this kind of application” rather than “what programming language is my favourite”. I like to be put in an environment, where I have to adapt, because I am always eager to learn. This project provided that kind of environment – I had never used Unity before, I had almost no experience in C Sharp and had no experience in game engineering. This 3D game is a great addition to my portfolio of work. In my opinion, we exceeded our own expectations and I am delighted that I was a big part of it. A great set of skills that I developed or improved throughout the project and the game itself are valuable things to showcase to employers, especially game development companies.

What would I do differently in the future

I am proud of my team and our product, which turned out into a playable fun game. I even think that with slight modifications and improvements we could turn it into a game that is worth sharing with the world’s audience of gamers. However, there are some things that I would do differently in the future. After dividing roles for each team member, I would spend more time to look at all my responsibilities and do more research on tools and functionalities I would be using to achieve my tasks. For example, we had an idea to implement upper body rotation while the player is aiming his weapon up and down. This task was one of the last ones, because it had to have a working character, weapon, movement, aiming and camera. So, knowing that I would approach this task at the second half of the project, I did not spend enough time at the beginning to at least have an idea what functions or tools I would use. Later, when it was the time to work on this upper body rotation, I realised that it uses different bone constraints and using those breaks the character movement. To implement that I would have had to change the whole character movement functionality, which would take a lot of time. Eventually, I used a different technique to implement the weapon rotation instead of body rotation. Even though it turned out quite well, I thought that it was my mistake and I should not repeat it in the future. Another thing that I would do in the future projects – I would suggest using a platform for the team’s tasks and time management. Typing out all the tasks, date when it’s supposed to be done and updating the progress would remove unnecessary confusion and help keeping the team on track. We basically did the same thing on Microsoft Teams meetings and our team’s Discord chat, but sometimes I needed to get information quickly and the teammate I needed was sleeping and vice versa. Keeping everything structured would help the team to cooperate more efficiently. The last thing I would do differently is make more mistakes. During the project I caught myself many times overthinking the problem, instead of trying to solve it in a way that I was almost sure would not work. Sometimes the solutions, I abandoned at first, worked perfectly fine, but even if they did not, I learned what works and what not, which is also an advantage.

References

Mixamo: <https://www.mixamo.com/#/>

Unity: <https://unity.com/>