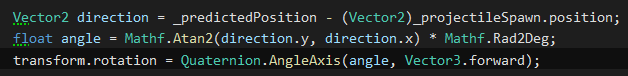
Reflective Statement:

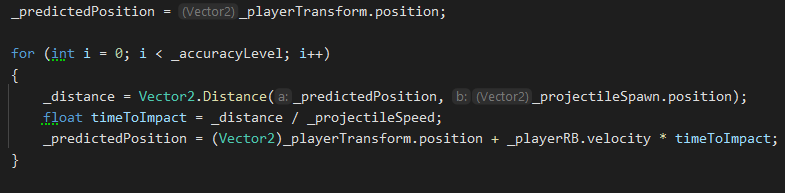
I spent a few years studying programming and learning Unity before working on these briefs. Since I am not new to programming, I challenged myself with three advanced programming briefs that I found interesting and useful for my future games. While working on these briefs, I faced various challenges, learned new skills, and gained valuable insights that will help me in future projects.

The goal of the first brief was to create a star system name generator to create 100 billion names randomly for a space exploration game. At first, I was thinking of getting a huge list of names. So, if I could have found 10,000 names, put them in a list, and gotten a random name 1-3 times. A name for a star could be something like John Jack Rob. The probability of that would have been 10,000 X 10,000 X 10,000 = 1,000,000,000,000 (1 trillion). However, my friend suggested I do something more interesting. Instead, all the star names would be numbers. For example, hundred fifty or one million two hundred thousand. This is the idea that I went along with. The way I ended up generating these names was by generating a random number from 0 to 100 billion and converting that number into a text form. The hard part was obviously converting the numbers into text. What I realised is that all the numbers consist of words zero-twenty, thirty, forty, fifty, sixty, seventy, eighty, ninety, hundred, thousand, million and billion. Just using these 31 words I can make up any number from 0 to 100 billion. All I had to do was to create a set of rules for the correct combination of words to generate upon feeding in a certain number. In the process, I learnt how to convert a string into a list of characters, using a type that I have never used before instead of int called long since the maximum value that an int can have been 2,147,483,647, which is smaller than 100 billion. The skills that I have learnt through this brief will be valuable when working with big numbers and strings.

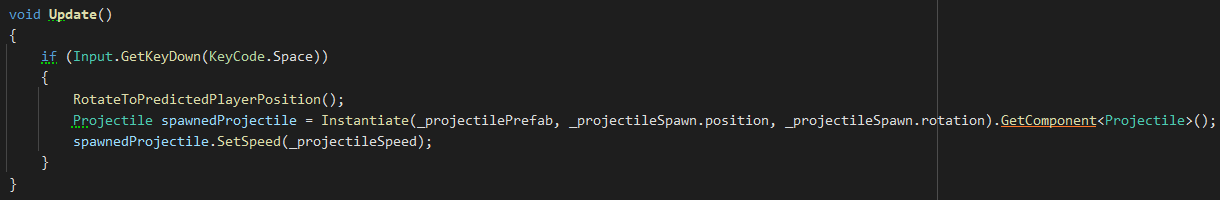
The second advanced brief I made was “Hit a moving target.” Firstly, I created a Player Movement script to move my player and a Projectile script to add a constant velocity to a projectile. Then I moved to the most difficult part of the project – the projectile prediction algorithm. The goal of the algorithm is to use the position of the bullet and its speed, the player’s position, and the player’s velocity to rotate the enemy at a perfect angle to hit my player moving target. The first thing that I figured out was how to aim the enemy directly at the player.



Then I worked out an equation to predict the future position of my player given the time it takes for the bullet to reach the player. However, after trying this algorithm, I noticed that the further the player went, the less accurate it seemed to be. The reason for this was that the distance between the player’s actual position and the future position is vastly different if the time for the bullet to reach the enemy is high and if the player is far from the enemy. Because of that, the enemy does not rotate enough to hit the player. However, this difference can be reduced upon iterating the algorithm multiple times. The more times repeat the algorithm, the higher the accuracy.



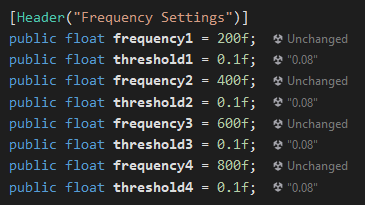
Upon pressing “Space” the enemy rotated towards the player and shoots a bullet. If a player was moving with a constant velocity, then the bullet will certainly hit the player. In the process, I have learnt how to put my thoughts on paper, which I have never done before because this problem made me remember my trigonometry. This will help when making enemy AIs in my games where AIs can shoot projectiles at my player to make the game more complex.



My third brief was to make an “Audio Frequency Detector.” It should be able to detect 4 distinct frequencies from the currently playing music track. Firstly, I created 4 cubes and made a script for all of them which can efficiently turn them on and off.



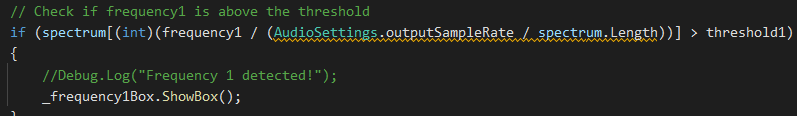
Then I created the actual “Frequency Detector” script. In there, I prepared the frequencies and thresholds to be detected which can be easily changed within the inspector.



To detect the frequencies, I have to get the spectrum data from the currently playing music track and save it into the array of floats.



Then I go over all the frequencies and check whether their volume is over the threshold set in the inspector, it is, then I show a cube associated with that frequency.



I had to do a lot of research to find this solution because other ways were a lot more complex and beyond my comprehension. I have learnt how make frequency bands in code and display in the Unity scene. This brief I found most difficult to complete since I knew the least about it in comparison to others. In the future, I can use this knowledge to create rhythm-based games or rhythm-based games I might be working on.

In conclusion, I am glad I had the opportunity to work on these briefs because they gave me insight into what it would feel like to work on tasks given by a higher-up or an employer. From generating names for a space exploration game to predicting the future position of a moving target and detecting audio frequencies, each brief presented its unique set of challenges that I would not be able to come up with myself. It taught me how to write documentation for my projects and this reflection helps me identify what areas need improvement. I recognize that there is still much to learn, and that programming is a continuously evolving field. Nonetheless, I am grateful for the opportunities to challenge myself and improve my abilities.