Skate Programmer Task

SYSTEMS

Movement

In this game I decided to do an arcade gameplay, very simple movement and mechanics. There is no fancy physics like in Skate 3, but it does do the job. I'm only using the default Movement Component form the Character class and added some friction to give the illusion of inertia.

Holding W will make the character accelerate until reaching the max velocity, and the animations for movement are dictated by the velocity of the character. If the character didn't reach the maximum speed, she will play the animation of speed up, but if reached the maximum speed, it will be played the animation of base skateboarding.

When no button is pressed, I decided to only play the default Idle animation, and I think it ended up being a nice choice.

For the jump action, I got a jump animation that was very close to a jump action that you would expect from someone in the top of a board. I used an Anim Montage to control when to really perform the jump action (using an Anim Notify). Unfortunately, I had to decide to have some delay in the actual jump to be coherent with the animation, which was the best I found.

Score

When performing a jump, I coded some Line Traces to detect if there are any Obstacles bellow the Character. If it detects an Obstacle and there are no collisions with the Character, it will try to compute the score depending on the type of the Obstacle.

There are 3 types of Obstacles: Small, Medium and Large. Small gives 2 points, Medium gives 4 points and Large gives 8 points. I made this to give some diversification in the points.

As an extra, I've put a timer of 2 minutes to define when the match ends. When the match ends, the game restarts after 3 seconds.

MY PERFORMANCE

I'm not really used to using animations yet, so it was a good challenge making this skate game.

I think the most complicated part was to align the animations with the gameplay, although it's not perfect what I did, I think it's acceptable for a 48 hour prototype, and that was what I was aiming for.

I tried to focus more on the mechanics part, as you have highlighted that I needed to show my C++ skills. Probably it would be better if I did a game more focused on physics for that purpose, but I wasn't confident in messing up with physics in this short time.

In the UI I used more Blueprints, but I mixed both C++ and BPs, because I wanted to show that I also know how to use both in a healthy way.

TIME SPENT

- **Planning and research:** ~ 3 hours
- **Gathering assets and making them work in the engine:** ~ 2 hours
- **Movement System with animations:** ~ 4 hours
- **Score system:** ~ 4 hours
- **UI:** ~ 2 hours
- **Level Design**: ~ 2 hours

Total time: 17 hours