Benjamin Pointer

Rocket Cop

Week 1 (23rd - 30th May)

This week I worked on a super simple trick system. I started playing with the rotation of the player's character multiplied by the direction of the analogue stick being utilized. Essentially this created a pretty fun game-style although it was near impossible. Sadly it would've been quite possibly the most efficient way to handle tricks but it would've just been too hard for the player. Here is a snapshot of the base which only rotated the z position of the player but I would've split it out based on the current position of the stick and applied rotation by that…

```
if (Input.GetAxis("LeftJoystickX_P") < 0.19 || Input.GetAxis("LeftJoystickX_P") > 0.19)

    {
     var vec3;
     vec3 = new Vector3(0, 0, (Input.GetAxis("LeftJoystickX_P")) * m_rotationSensitivity);
     this.transform.Rotate(vec3);
    }
```

I then decided to split it up like my original idea, which was to utilize the direction of the analogue position provided the player wasn't grounded to perform some form of trick. I animated the character in a variety of ways and then utilized those animations based on the player's left analogue stick.

Week 2 (30th May – 6th June)

This week I fully revised the controller script. It originally performed ollie/nollie tricks extremely inaccurately – you literally could just hold up and you would consistently ollie. After fixing this I had to tweak the system a bit for the tricks to work in a manner that didn't annoy the player, which lead me to create a pause screen and implement pause on death. There was a heap of other stuff that I worked on this week, just a majority isn't worth noting. I completed a lot of bug and previous error fixing.

I worked on a combo multiplier system which rewards the player for tricks completed, it is then multiplied by the amount of tricks performed within the current ollie/nollie state which only increases if the current trick is different than the prior trick.

I implemented a hit-box which is rectangular in shape and is placed above/on-top of the player-vehicle, this acts as a collider for when the player attempts a trick too late and thus the character is being animated whilst being on this ground – this will now cause the collider to collide with the road object and thus result in a crash and ultimately the game over scenario.

I also implemented the rotation of the vehicles wheels by script as opposed to the unity animation layering system because I'm really lazy.

Week 3 (6th – 13th June)

This week I worked on a cheat system and implemented controller controls within the main menu. The cheat system is super basic and is initiated by pressing the 'L1' button followed by an array of inputs entirely from the 4 buttons available. Some of the cheats that I implemented this week include an

Throwaway Games

increased power-jump which boosts the player almost three times higher than normal (L1,B,A,B,X) and another which gifts the player an entire minute of time (L1,A,A,A,A)

I also sought after some super basic sounds and implemented them through my simple PlaySound function which utilizes one audio-source located within the Player's game object that load's multiple clips and play's upon action.

I also implemented Nick's screen-shake script to play on landing after jumping and on impact from collision obstacles.

Feedback provided from the lecturers led me to re-map the controller setup so that it was less of a 'southpaw' orientated style.

Week 4 (13^{th} – 20^{th} June)

I did a heap of simple fixes and touches this week. I worked on rewriting Blair's animated popup system – so that it's somewhat more optimized in the sense that it utilizes one object and then alters the text dependent upon the situation rather than running pre-defined objects assigned with text. This was utilized within the cheat system as a form of feedback to notify the player the status of the cheat being entered.

I implemented two new cheats, one of which slows the game down giving a slow-motion affect (L1,A,Y,X,A) and the other decreases gravity so that the player returns to the ground slower thus meaning they are in the air for longer and can perform more tricks (L1,A,X,X,A).

I also implemented some changes which the lecturers had told us to do, such as the alternating game-over scene which just changes the text depending on whether the player has crashed or whether the timer has ended.

I looked into implementing a hit-box multiplier to spawn above object so that when a player tricks over an object it would give them an increased multiplier – I came to the conclusion that if we weren't splitting the modes (having a smash-em-all and a trick mode) then it wouldn't be worth it as there is so many objects being spawned and thus it'd be somewhat simple.

Throwaway Games