Project Phoebe

Game Development Journal

Kyle Jussab

The original idea

I’m the biggest fan of Naughty Dog games. I want to make games similar to the style of games that they make. As part of the Unity Junior Pathway we’re introduced to some basic elements of Unity, and we’re finally introduced to scripting. This is more in line with what I’ve been learning in my Bachelor’s, so I’m starting to feel very confident about the project. The lecturer introduces me to a Project Design Document, and how to write one for the project in this pathway. I immediately think of making a 3rd person shooter. It doesn’t have to be super complicated, all I need is the player to shoot something, and if it hits an object destroy it. I could have enemies standing and firing shots back at a much slower rate for a little but of a challenge. Destroy all enemies and get to the end of the level. That was the idea, something to get me used to how 3rd person shooters work at a fundamental level.

Here’s a copy of my Project Design Document.

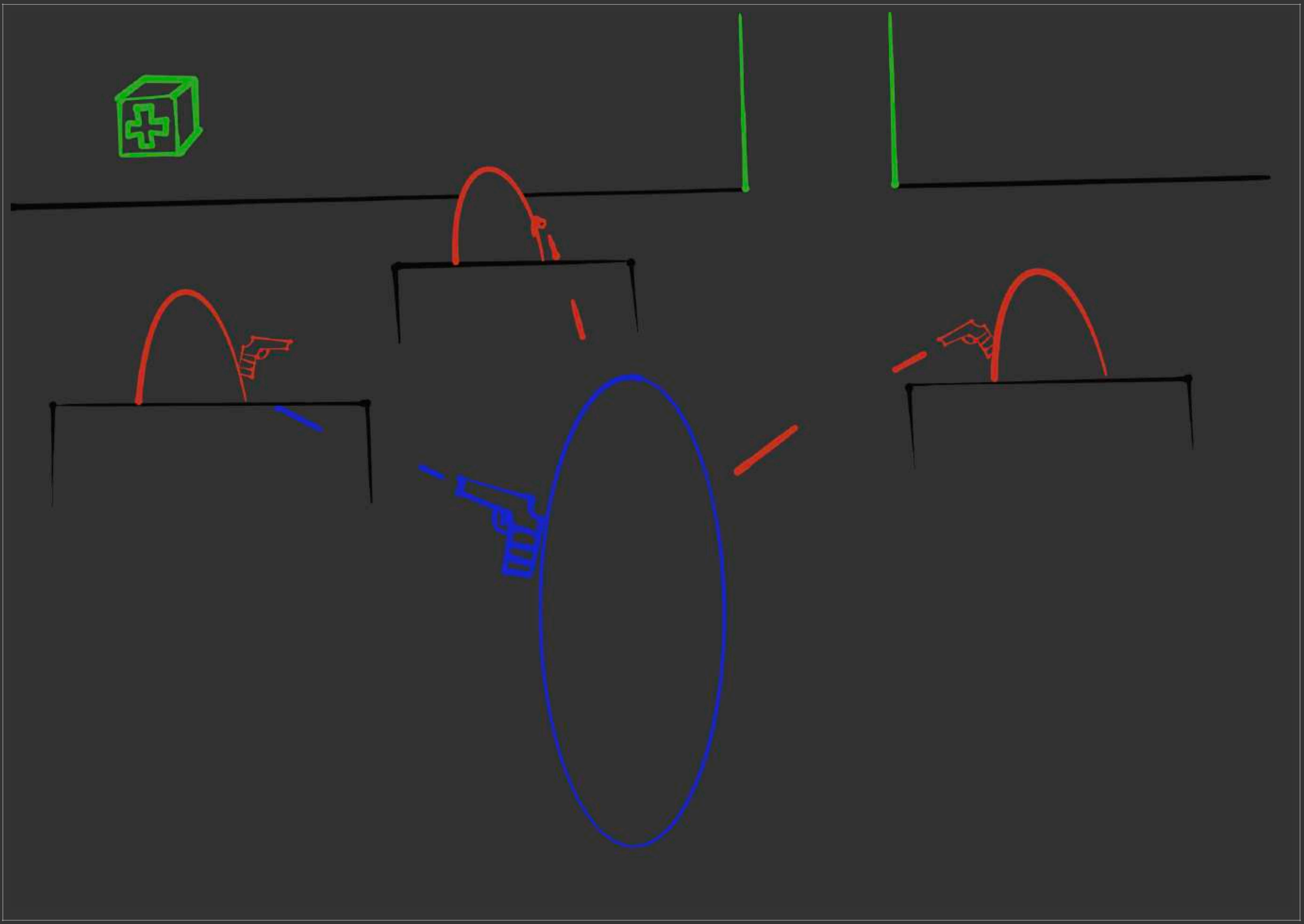
Project Concept

1. Player Control	You control a **human protagonist** in this **third person** game where **controller input** makes the player **move and shoot projectiles.**
2. Basic Gameplay	During the game, **enemies (human or otherwise)** appear from **in front of you** and the goal of the game is to **get to the end, and take out as many enemies as possible.**
3. Sound and Effects	There will be sound effects **when you shoot and hit an enemy**, and particle effects **when a shot hits you or the enemy.** [optional] There will also be **a sound effect when you’re low on health, and a visual indication.**
4. Gameplay Mechanics	As the game progresses, **stronger enemies appear** making it **harder to get to the end**. [optional] There will also be **a crouch mechanic, allowing you to hide behind cover**.
5. User Interface	The **health** will **decrease** whenever **you get hit**. At the start of the game the title **“Shoot and Run”** will appear and the game will end **when you get to the end of the level**.
6. Other Features	**A way for health to be regenerated. Shooting a health object or gaining it from dropped enemies.**

Project Timeline

Milestone	Description	Due
#1	- Project / Camera set up with primitives for all gameplay objects	13 Dec
#2	- Player can move and not leave play area	20 Dec
#3	- Player can shoot, enemy can shoot, both player and enemy takes damage	27 Dec
#4	- Primitives replaced with 3D assets, and UI added with health tracking	3 Jan
#5	- Sounds added, particles added, and main menu	10 Jan
Backlog	- Health crates or health regeneration - Crouch - Power ups	17 Jan

Project Sketch



The Problem

I bit off more than I can chew. I got too wrapped up in whether the animations would work, or if I had the right 3rd person camera, and it got overly complicated very quickly. I'm new to Unity and it's tools, I'm new to game development, and I'm a programmer. I shouldn't be worried about how it looks, let alone with getting the best camera and systems. The idea of this project was to take all my junior knowledge of Unity and make a project, but instead I gave up on the project on the 3rd milestone. Eventually I will be able to make a shooter, with slick cool animations but that is not now.

That is important! It was stated by the lecturer and though I tried to keep my ideas small to begin with, they started to balloon off into tasks beyond my current capabilities. Though small in my head, it still requires a lot of work and I didn't want to sacrifice more of the general idea of the game. So I gave up, I didn't want to make the game if I couldn't even make it look and feel like half of what I had originally planned. It's beyond my capabilities right now, but that doesn't mean it will be forever. I skipped way too many vital steps. I need to get good at being a junior before I can advance to being intermediate. That's exactly why I fell at the very first intermediate challenge.

You might ask why I wanted to get that far into the game (when it's my first Unity game) anyway. I really want to start working for Naughty Dog, and I feel like I'm far off everyone else applying. I need to have GOOD projects if I want to stand a chance of getting a job there (which believe me, I want to give myself every possible chance).

After the slump I slowly realised that what I'm asking of myself is more than I can handle, and that's okay. My goals might have been a little unrealistic to begin with anyway. How would I possibly get into one of the most talented game studios, in the space of 2 years, with which I had no game development knowledge at the start of that timeframe. It also settled easier in my mind knowing that even if for some reason Naughty Dog did hire me, my contributions wouldn't be good at all. What's the point of being in Naughty Dog if I can't add to the talent? Let me gain the talent first, that way I can make meaningful contributions, like the ones that they made in the past that inspired me. Meaningful games, not meaningless ones.

Project Phoebe

I found confidence in the project by switching it from a shooter to a puzzle game. My idea was inspired by seeing a lot of beginner games having a very simple colour scheme. There are no complicated shapes, just primitives, with a simple colour palette. These games looked good! They didn't need to have a human character; you could have a sphere that had a personality, and it worked! It took the complexity of animation rigs and all that jazz out of my first project. Let's build the fundamentals of a game and understand how it works, let's make it fun, and see if my idea has enough legs to justify making more levels and building a story around it. Let's get the "core" experience down.

Removing shooting was essential. What if we have no enemies? That stops me from having to worry about AI. Okay… but now what is there to do? Well what if to get to the end we need to solve puzzles that open doors to more puzzle rooms, and then once all puzzle rooms are done the level is complete. Okay… but "puzzles" seems like it could get a little too complicated too quickly, and we don't want to give up on this project again. What if it was one mechanic, like stepping on a pressure plate, and that opens the door? Okay seems a little easy so maybe we could add some basic platforming to place multiple pressure plates in different places. Doesn't really make the game harder, but that's okay, it adds progression, that way we have to do more than just step on a single plate.

It also has potential for improvements with iteration. What if other objects can activate pressure plates? What if you could grab and throw these objects? What if the plates are on the wall or the ceiling? What if there's something else in the room that you have to work around or deal with first? Whoa.. that one seems cool… what if eventually those things are enemies that shoot at you… that you need to shoot at? This is starting to get cooler and cooler, and it's not necessarily something I need to implement now, it's something I can worry about when I learn more about Unity. I mean think about it, what if there's a timer for each room, and not only do the enemies shoot at you, they can deactivate the pressure plates? Super cool, this is starting to sound like game, and all I have to worry about is two objects colliding, and when that happens a trigger is sent to raise the door. I can do that, that's within my capabilities right now. I can challenge myself by adding more plates, and more than one room in a level, and all my objects can stay as basic primitives. Boom, we've got an idea, more importantly, an ACHIEVABLE idea. This is where I got the idea to call it Project Phoebe. The Greek titaness was the goddess of bright intellect, and though I wouldn't call the project particularly intellectual, my approach to this project changed and that changed how I approach future projects. I understand a little bit more about myself and game development, at least in terms of building systems. Project Phoebe it is!

So let's redo that Project Design Documentation.

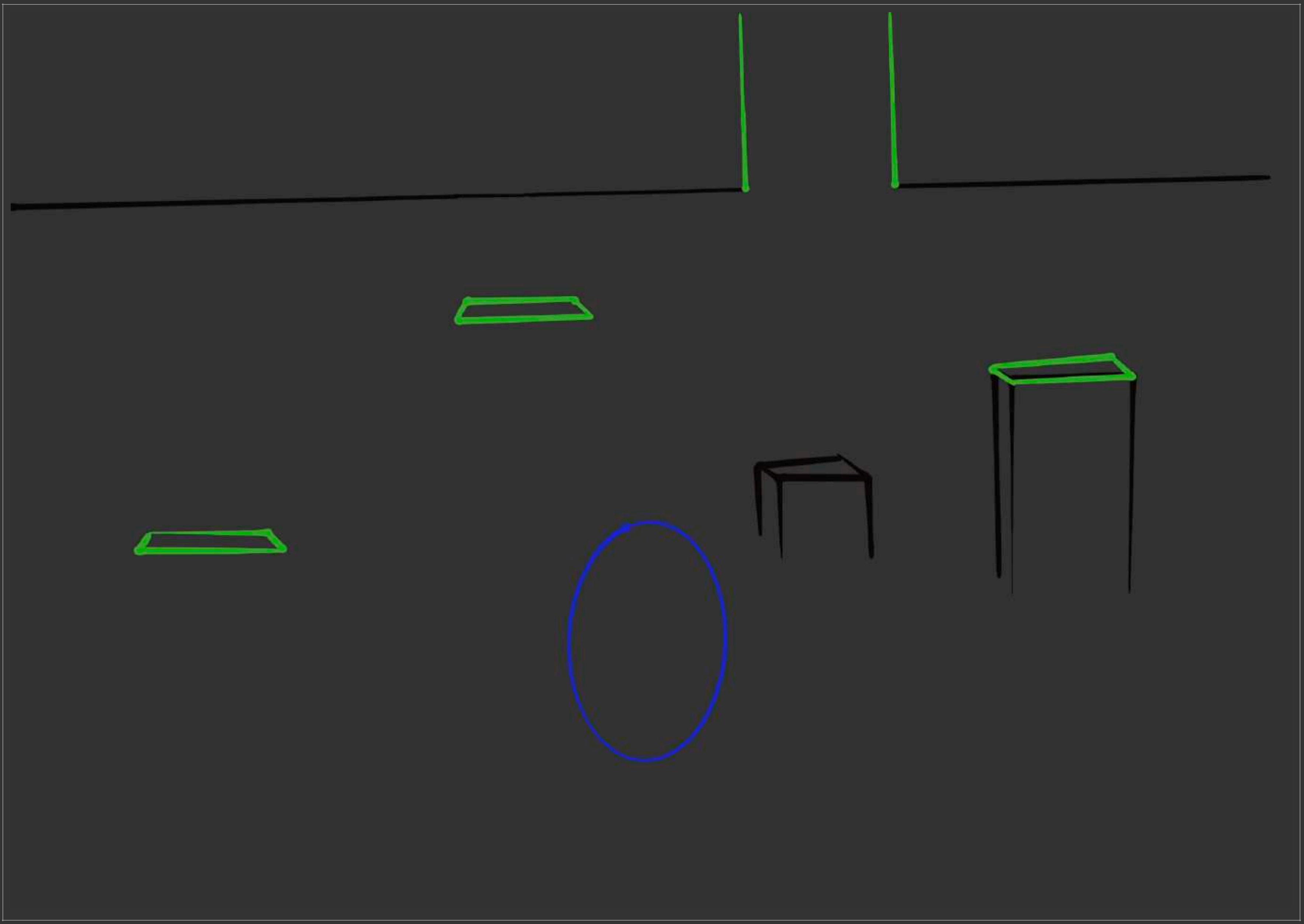
(New) Project Concept

1. Player Control	You control a **protagonist** in this **third person** game where **controller input** makes the player **move and jump.**
2. Basic Gameplay	During the game, **pressure plates** are placed **around the room you're in** and the goal of the game is to **step on all pressure plates to open the door to the next area (or till you get to the end).**
3. Sound and Effects	There will be sound effects **when you step on a plate**, and particle effects **on the door when you activate all of the plates.** [optional] There will also be **a sound effect when the door opens.**
4. Gameplay Mechanics	As the game progresses, **there will be more platforms in more challenging areas** making it **harder to get to the end**. [optional] There will also be **an object that stops you getting to a plate (such as a spinner).**
5. User Interface	The **time** will **decrease** whenever **you enter a room**. At the start of the game the title **"Project Phoebe"** will appear and the game will end **when you get to the end of the level**.
6. Other Features	**Tutorials. A way of showing the player how to play, and if other objects were added, what those objects do.**

(New) Project Timeline

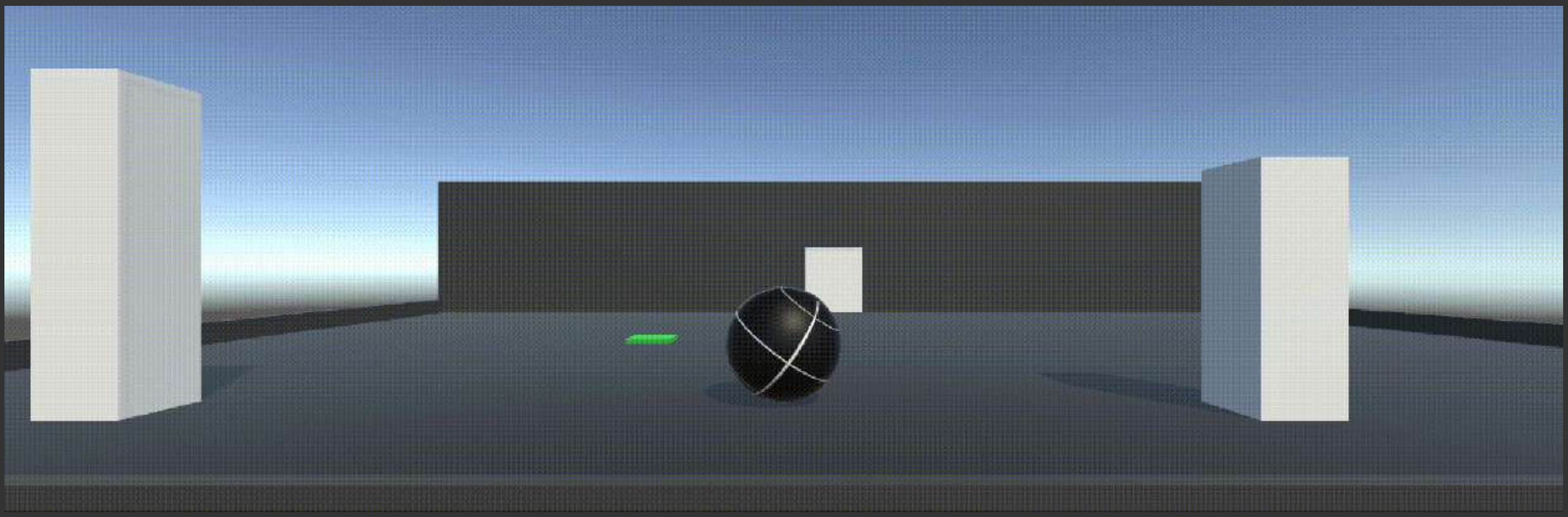
Milestone	Description	Due
#1	- Project / Camera set up with primitives for all gameplay objects	27 Mar
#2	- Player can move and not leave play area, pressure plates trigger event when touched. They remain activated. Once all pressure plates are activated a door opens.	3 Apr
#3	- Level design. A minimum of 3 rooms are built, with progressing difficulty. The game at this point plays exactly how it should play in the finished project.	17 Apr
#4	- General feel of the game. Colours can be changed, how the UI is displayed and so on.	24 Apr
#5	- Sounds added, particles added, and main menu	1 May
Backlog	- Spinner - Tutorials	8 May

(New) Project Sketch

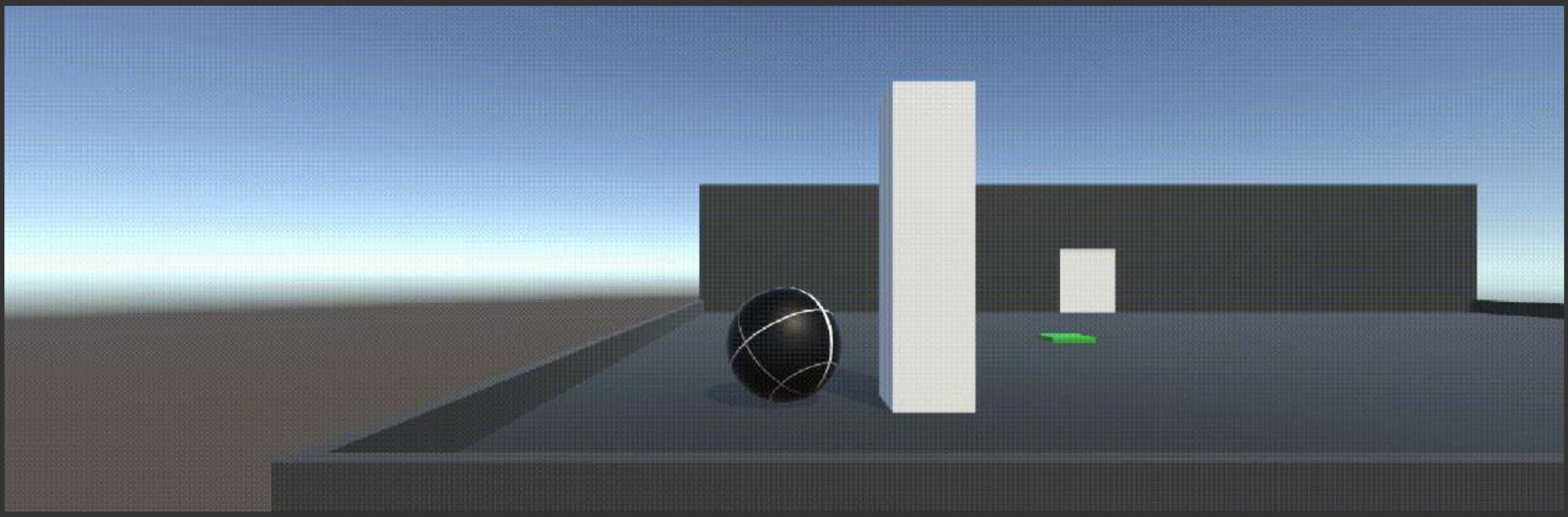


Discovering the game

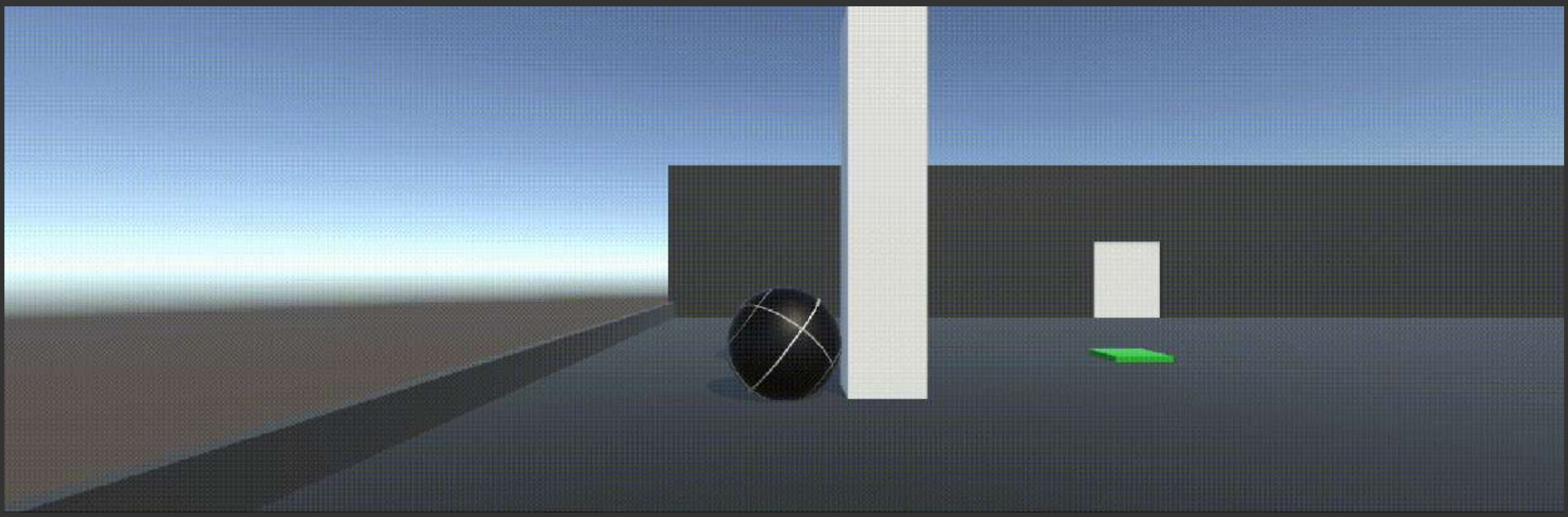
As I developed the game further, I found that just playing around in the game was fun. I know that I'm just moving around a sphere, but I made that sphere move, with my PlayStation controller! That's so cool! As I was saying, in the fun I had I found some flaws in the physics and the way I coded the movement. I was able to mantle platforms.



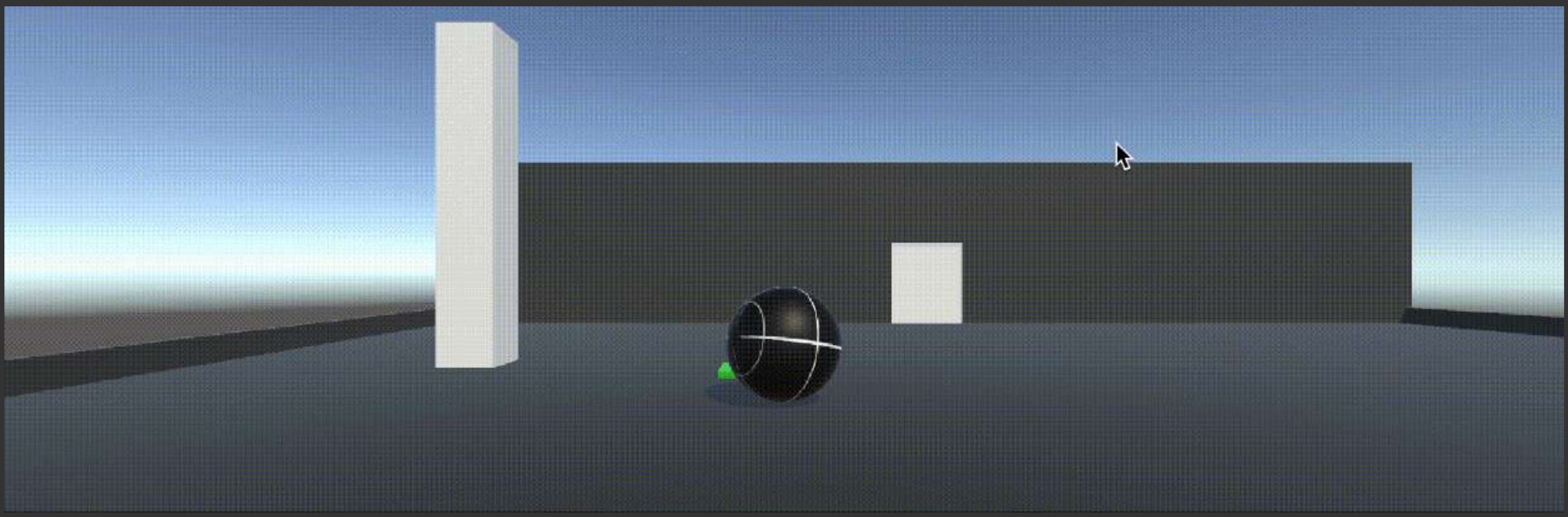
I was able to bounce off a small wall to go higher than I could with a mantle.



I then asked the question… can I combine the two? And well yes… yes I can.



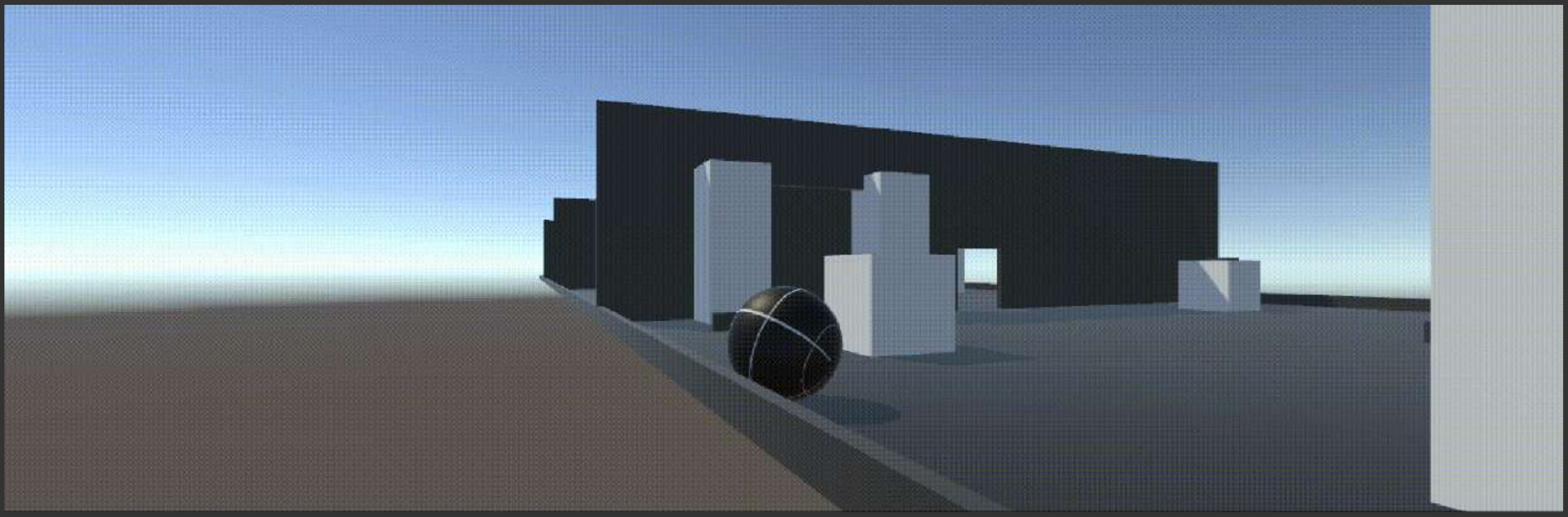
Now the question is just how far can it go? With a little momentum it can go even higher!

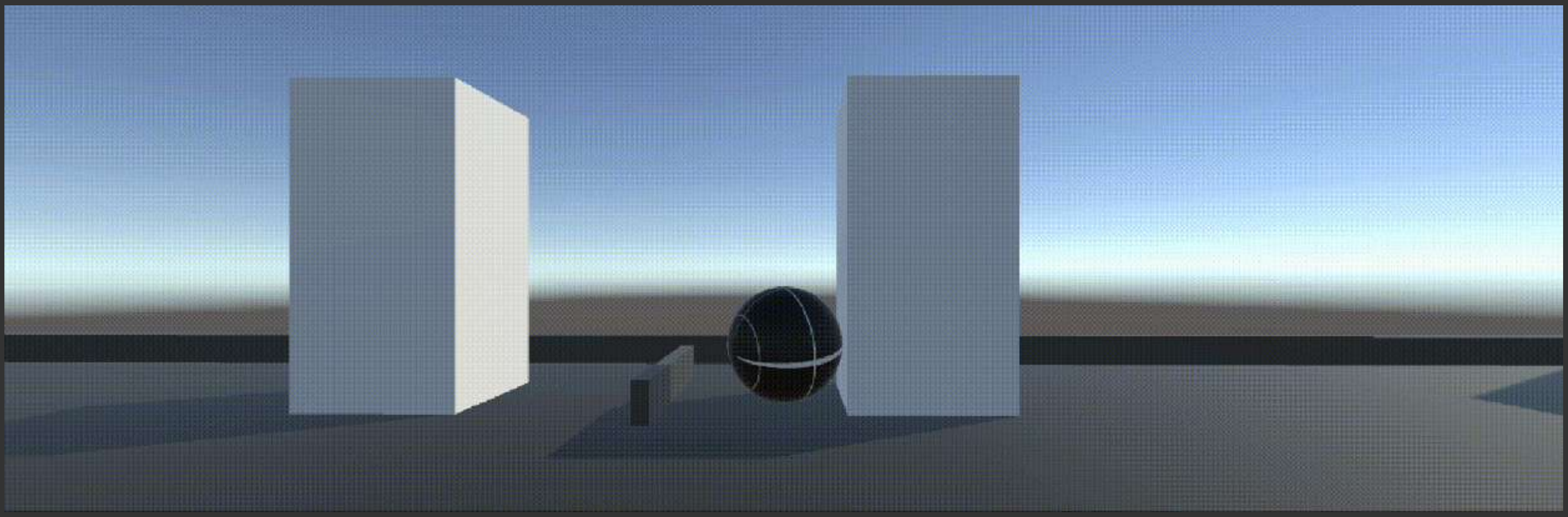


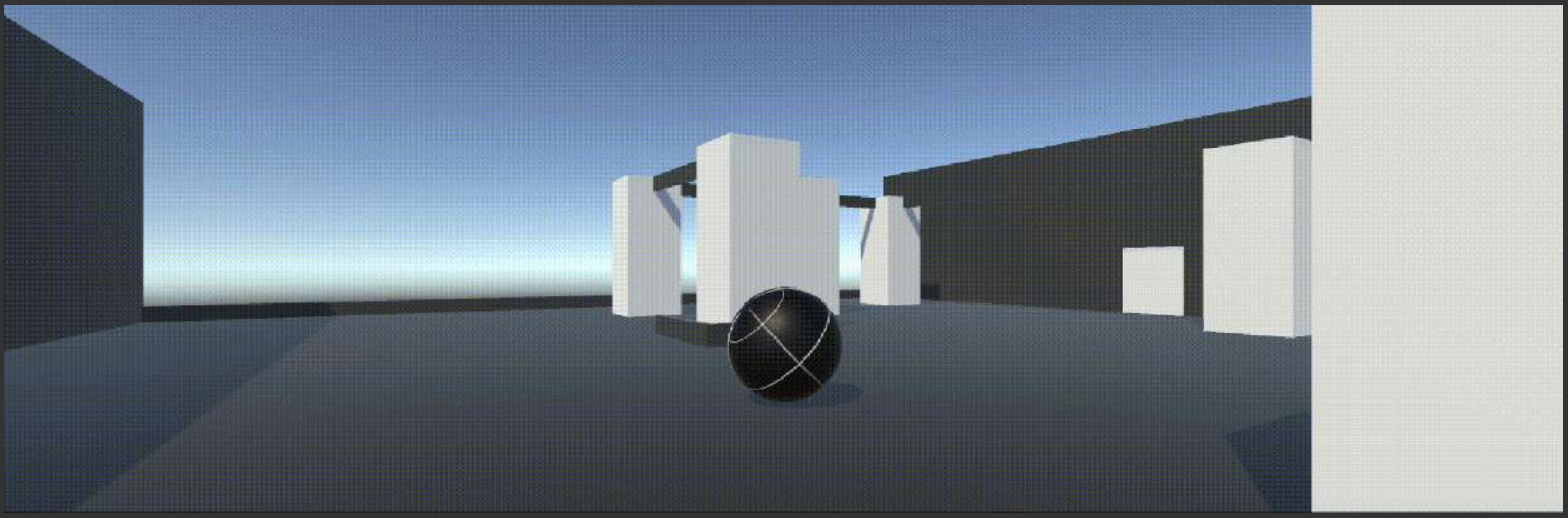
Each step up from here is a genuine improvement on skill. I found it really hard to do the momentum + high jump + mantle, but I once found the high jump hard and now I can easily do that. Having discovered something small like this during development, means I can now add a skilful section to the game. I also like the fact that once on a pillar you can no longer jump. The idea is to get higher you need to jump higher; now with the various stages of jump, I can create slightly more challenging levels.

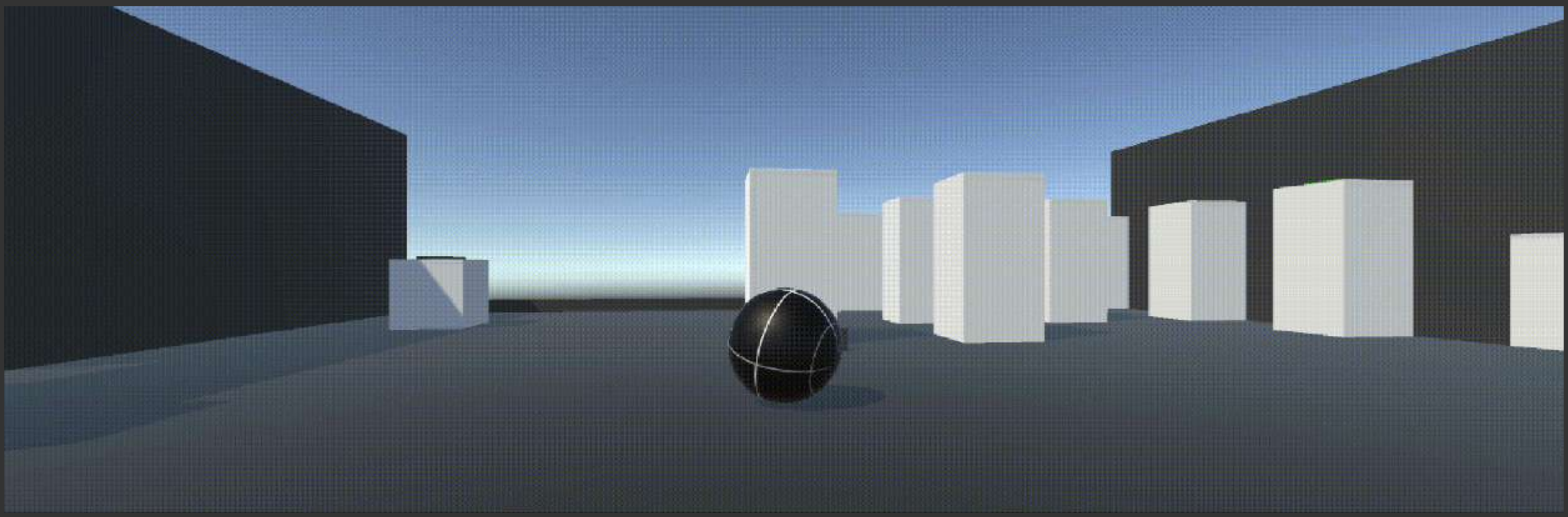
Level design

As the main constraint for this game was the fact that I can only jump on the ground (I didn't want to be able to jump off pillars), I wanted to find creative ways to make different challenges with such a constraint. Exhaust all the possible things I can do with being able to jump higher by bouncing off a boundary. A lot of what I discovered was the more momentum you have the higher your bounce upwards can be. So limiting your momentum with how high the pillar is invited challenge. I played around with a few more things, like putting barriers in the air, and making more "platforming" challenges, and not just jumping ones.







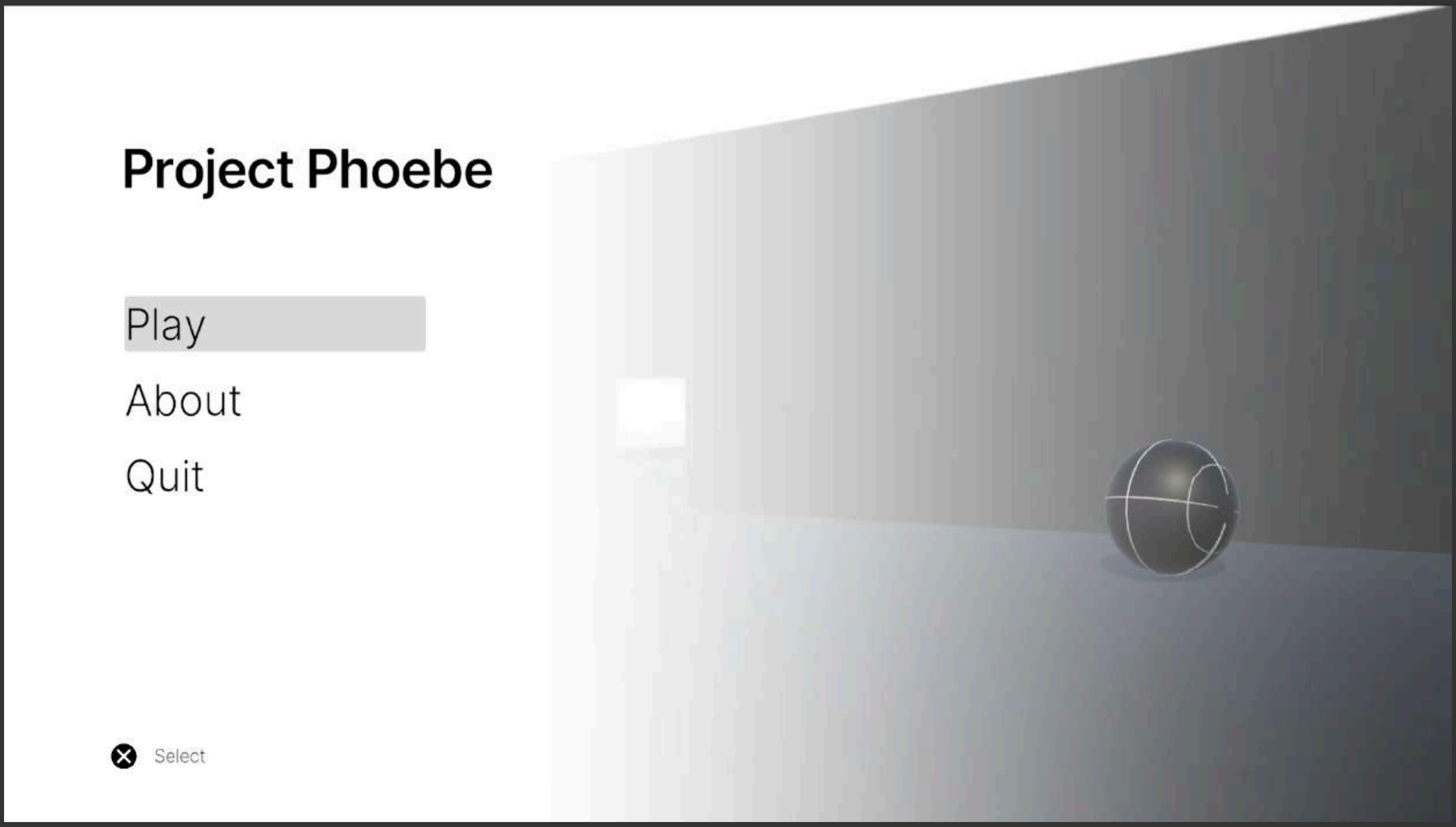


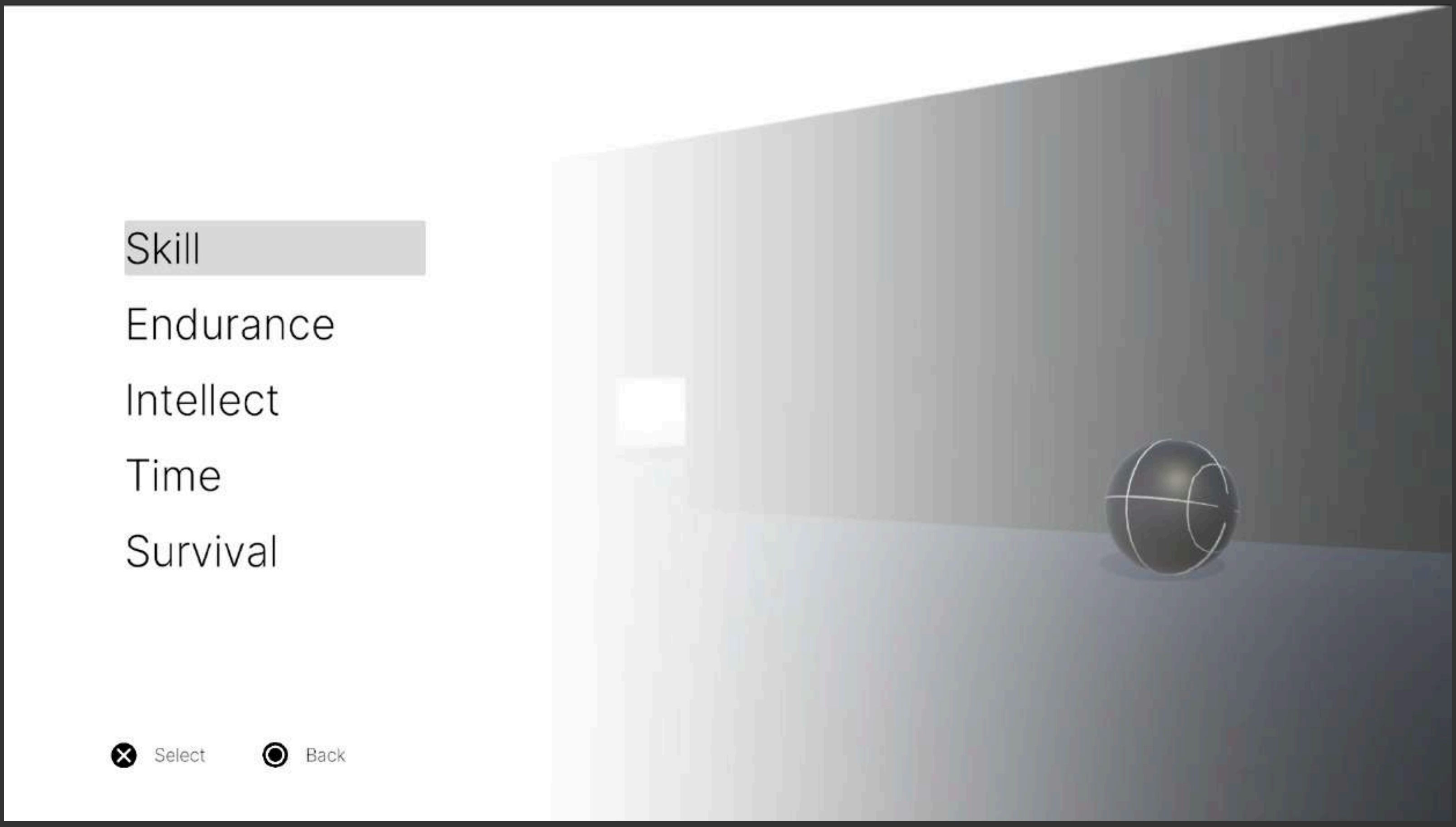
Once I completed all stages I realised that I was able to make a playable section without any of the enemies I originally wanted. It was still relatively challenging and it was fun as it felt like through each section your skills with jumping improved. I still have time to add enemies though… so why don't we call this a level? With the focus of this level being "platforming", and we make another level with the focus on getting around the AI? Or even defeating them? This idea sparked the potential to add more levels, in different scenes with different criteria on how to get to the end. I thought of 4, with their titles being as follows: Skill, Endurance, Intellect, and Time. What I've just completed is "Skill". Endurance is about dealing with enemy spheres!

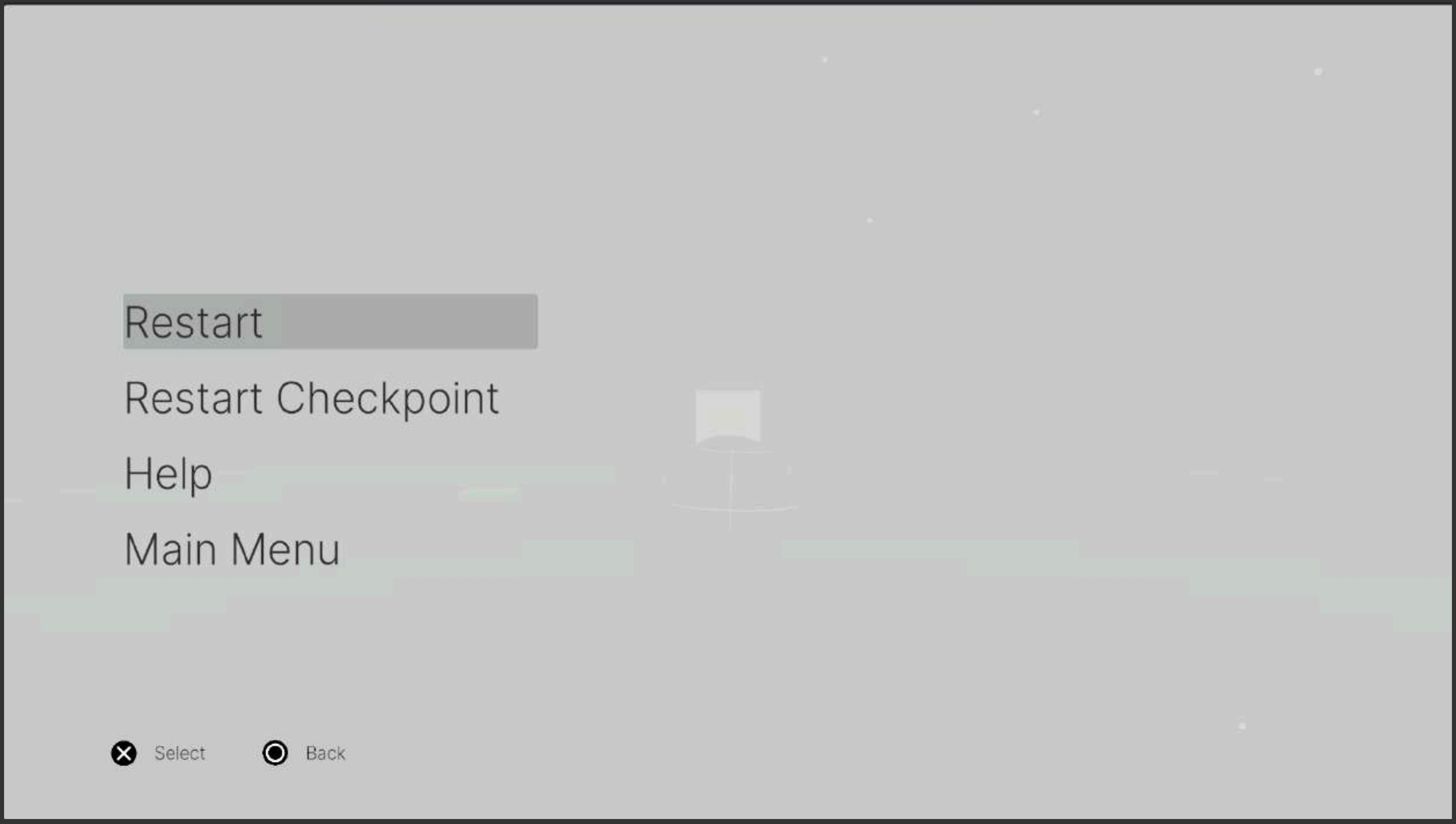
Making it a game

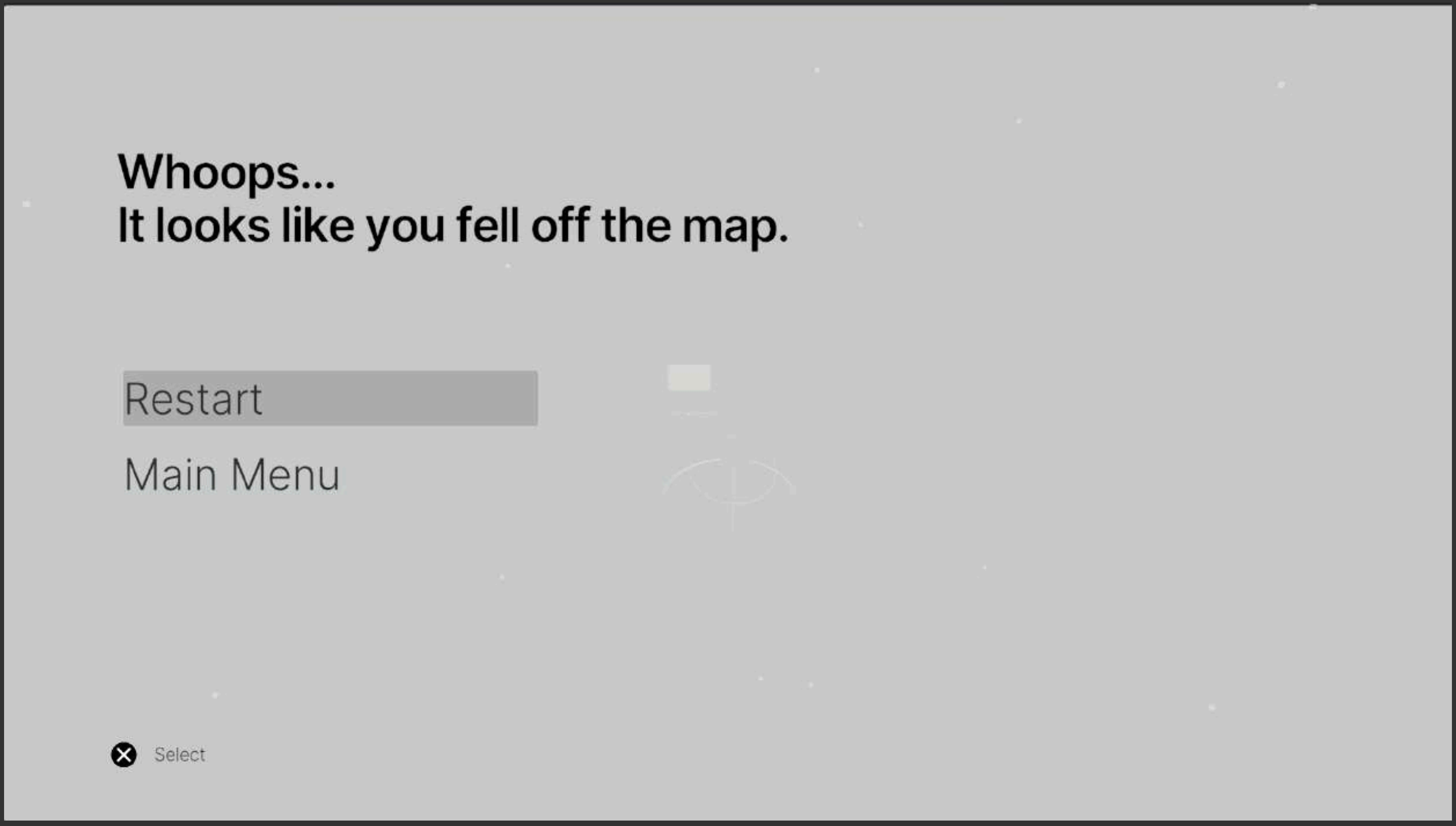
Spoilers… I'm not going to talk about Endurance right now. I actually really wanted to make the game a "game". So before I went on, I spent some time putting together a main menu, pause screen and a game over screen. I thought that each "level" beyond this point would have game over screens, a pause menu, some general help on the game, so I might as well get it knocked out of the park from the get go. Managing the state of the game isn't actually that hard, but I suppose that I worried about state quite early on in development. I implemented a very choppy "restart from checkpoint" button. I say choppy because I mapped the submit key to jump, and with my current code every time you restart from checkpoint the sphere jumps. There most probably is a more efficient way of handling save states, I currently just have the game manager remember the room the player is in from triggers, and when the button is clicked the whole level is reset and places the player in the saved room with all the relevant plates activated. As there are only 7 room and under 20 plates it was easy to manage. If I was to build out a multi stage system this is a very inefficient way of doing it. However, I didn't Google how to do it, I came up with the solution on my own, which I'm very proud of.

In knowing that I would make each level separately, that also means each level has it's own "game manager", which defeats the purpose of a game manager if I'm honest. Maybe next time I'll use multiple level managers, and keep with a single overall game manager. This approach means I have to make pause and save states for every single level. Looking ahead, I can see that this isn't the best solution either. I will push through, as each level is quite small… but next time, knowing what I know about how I've handled UI and menus… I'll do better. It is a learning experience but I'm happy that my first year of university in computer science led to me handling the documentation better, writing more modular and cleaner code, framing my mind in a state where I can figure out a solution on my own, figure out why the solution I came up with isn't the best, and more importantly how I can eventually go on to find a better solution. Skills are transferable and even though I learnt these techniques outside of game development, software is still software. Thinking forward this is very encouraging. When I first learnt these skills I didn't think they'd be applicable that quickly. Though I started coding in p5 with javascript I was easily able to transfer to C++ in university and now C#, and everything MAKES SENSE. Anyway, I'm getting ahead of myself, below are some images of my menu. Heavily inspired by how Naughty Dog handle their menu's, I thought not only is it minimal, but it serves it's purpose. A homage to them in a way, though there will definitely be more inspiration taken from not just them, but many other game devs in my games.








Whoops...
It looks like you fell off the map.

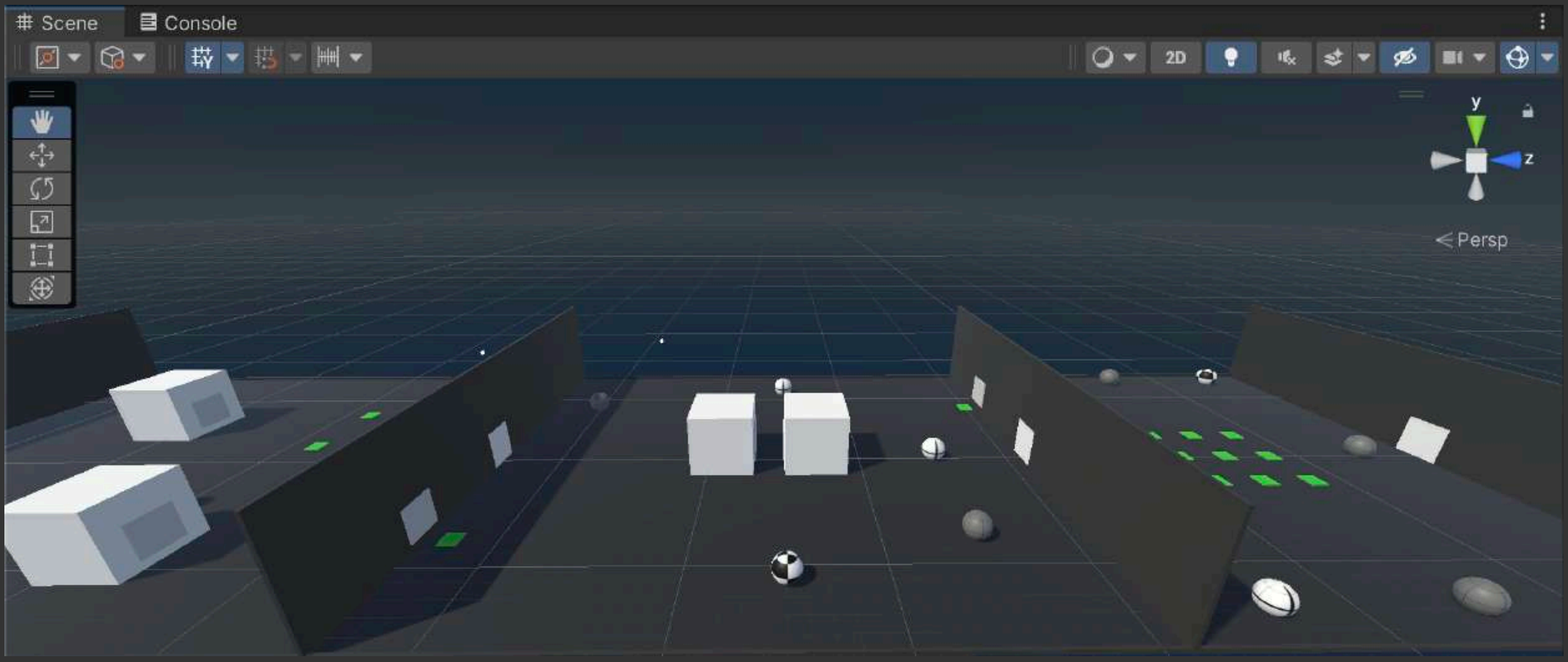
Endurance

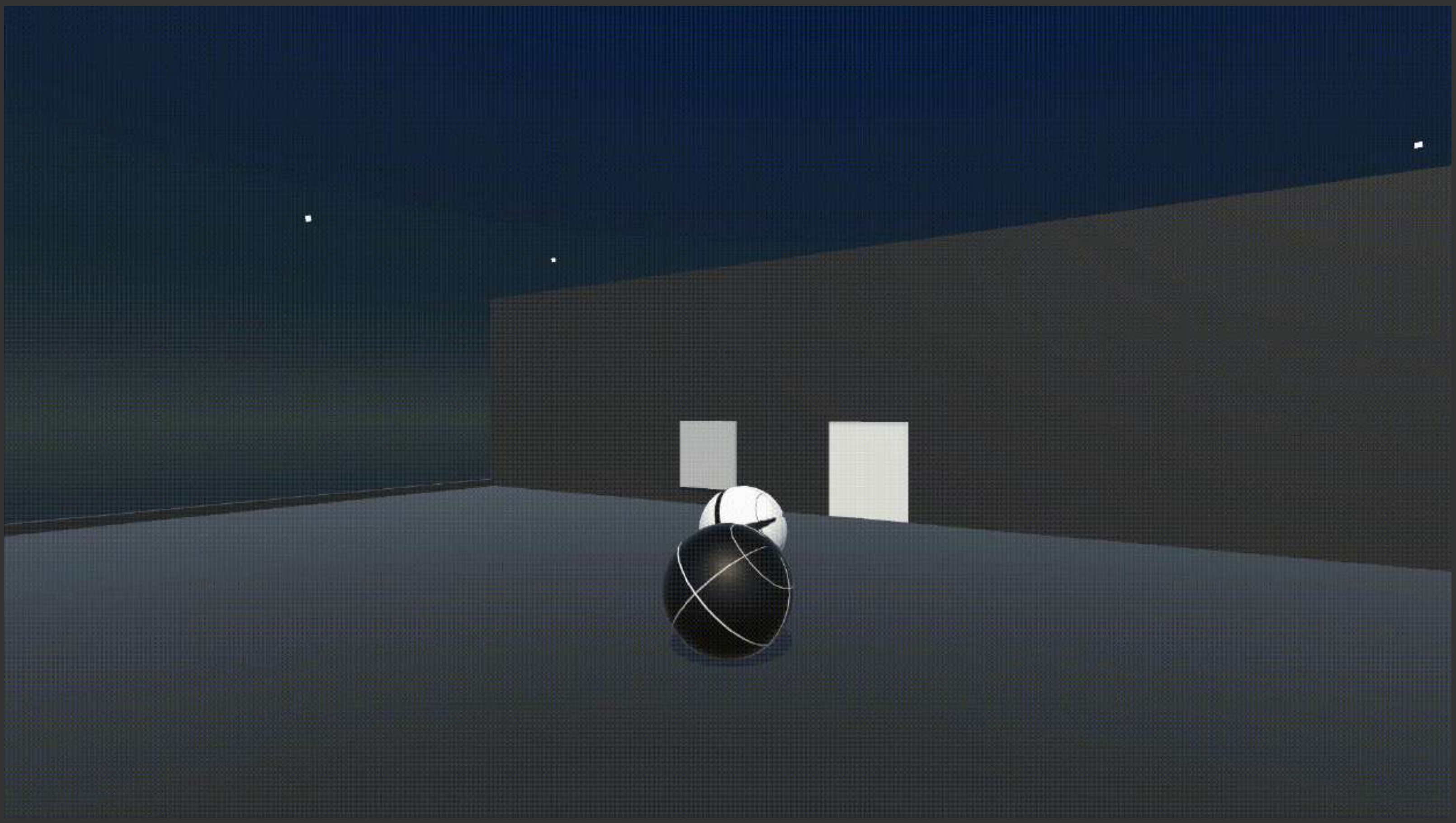
Endurance! Finally right! I decided on 3 enemy types, one that pursues you indefinitely, one that pursues you periodically, and another enemy that doesn't pursue you at all but deactivates any activated plates. This was also an opportunity to add a wall pressure plate, and I decided that only enemy spheres can activate wall plates. This means I needed to add a "bounce back" force to the enemies when they collided with the player, it also added a little variation to what was an already familiar game loop from Skill. I kept the player activated plates but originally had the 'deactivating' enemies only deactivate player plates (and not the wall plates). This made them a very useless enemy, the player wouldn't have to worry about that enemy at all and can deal with them last. I later decided that they can deactivate all plates, and now they are the most important enemy. The player has no choice but to remove these enemies as soon as possible despite them being the slowest, forcing them to avoid pursuing the faster enemies making it more challenging and a lot more fun.

As I wanted the enemies to be destroyed when they fell off the map, I naturally added a script that detected that. There was a very big problem with this approach though. For the most part because enemies can unintentionally fling themselves off the map that would leave the player with unactivated enemy plates, with no enemies to activate them, so no way of progressing. The restart checkpoint is very useful in this situation, and though I don't like game mechanics that force the player to restart or reset it was needed in this case. The thing is… how can you reposition that exact enemy if you deleted the game object? I chose to spawn all enemies on start, and not spawn them during runtime; I was not instantiating anything. My fix was to keep track of what section the enemy is needed in, and if the player was in a section greater than that, then it's safe to delete that enemy. If the player's section was the same as the enemies, rather than deleting the game object I moved it to an area off screen. When I call restart checkpoint any enemies not on the map are still available, but once you progress and gain checkpoints, you don't need the previous enemies anymore.

I was actually really proud of that solution, considering I built levels with a specific layout of enemies and chose to not do any spawning at runtime. I would imagine bigger games have a similar method of dealing with enemy spawns, putting them offscreen but spawned, and just reposition them when needed.

Outside of that challenge, the level design took the same approach from Skill where I gradually increased the challenge. I did remove the ability to jump, I felt it wasn't needed in this circumstance. Other than that I think it's a pretty cool addition, definitely a different challenge of getting through the rooms, it has no platforming elements in it whatsoever but still feels like the same game.





A quick time check. I set out to have the player and general gameplay working by April 3… it's currently March 31 and I have 2 full levels built out, with different mechanics in each. I also have a main menu and sound set up so it's safe to say I gave myself more time than I originally needed. That's okay though, I took up this project between my university semesters so I could spend full days working on it.

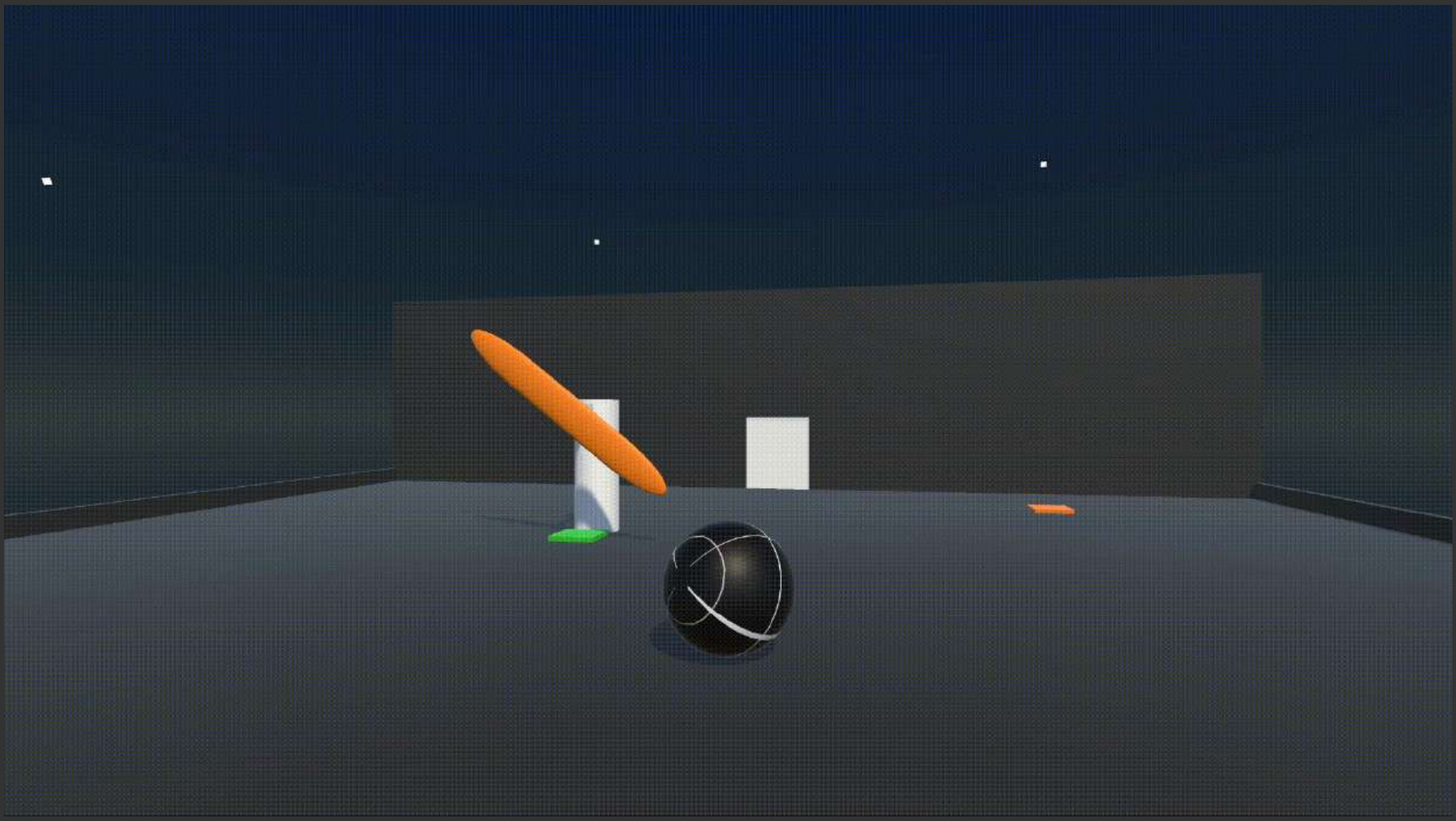
Intellect

Intellect being the puzzle level really posed some interesting questions for me to figure out. How would you build a system that remembers the order in which the pressure plates must be pressed? Not conventional at all, but I did it by having an array that held 3 values (for a puzzle that only had 3 pressure plates) and all values were set to 0. When the correct plate was pressed, the first value was changed to 1, in fact when every correct plate was pressed the following element in the array was changed to a 1. If at any point you activate a plate in the wrong order that array would get set to all 0's again. When the array was full of 1's the puzzle was solved. Though there's nothing wrong with this concept, the method I used to get the array to populate with 1's was terrible. There's no push method for arrays in C#, or at least I don't think there is, so fundamentally… using an array as the data structure for this was probably a bad idea. I stuck with it, and got it to work, but did not include anymore sequence puzzles for the rest of Intellect.

The rest of the puzzles were built around trying to get to the "a ha" moment of realising you can push spinners when they are inactive, and that inactive spinners can activate their corresponding plates. I think the solution is too easy though, after having built it out entirely I think when you get to that section the answer is a little too obvious. The final section has some “dud” plates and spinners that don’t do anything, and I think maybe adding those throughout the level would have made it a little more challenging to solve. Maybe even plates that activated and deactivated spinners, but had no relevance to the actual solution of the puzzle.

Puzzle design is not easy, in many ways you want to give the player enough information to figure it out, but not too much that the puzzle is trivial. I do not feel I struck that balance at all, I feel I made it too easy. Besides the AI it was the most fun I had developing a level, so puzzle design is definitely something I want to improve in, and scatter throughout any game I make even if it isn’t directly a puzzle game. I’m taking a lot of inspiration from traditional puzzles like sudoku and variant sudoku’s, and I think that at some point I’ll be able to craft a unique puzzle, that doesn’t feel like it’s been done else where. Making a puzzle isn’t about copying other puzzles!

Overall I think for a genuine first attempt this isn’t bad. What I’m more concerned with is the system that allows for puzzles to work the way they do. I will learn game and level design for coherence but inevitably I’m not going to be a designer. My job is to make the systems that the game runs on. I will wholeheartedly focus on programming before I consider making “fun” games. I think just having a puzzle system in place would mean that when I do get better at design I’d be able to just make puzzles, not thinking about building them and making them fun simultaneously.



Time

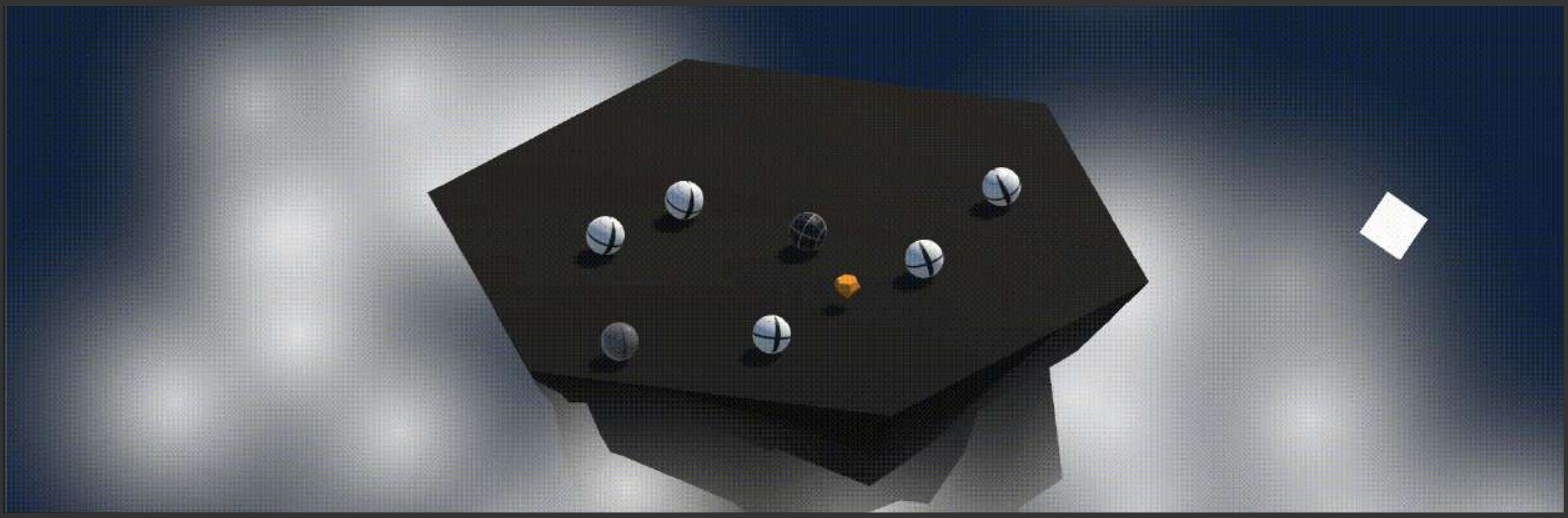
The time mode is very simple, the essence is to get to the end. All I added was a timer. Originally I wanted to make it quite complicated, and maybe incorporate some of the elements from Skill, Endurance and Intellect but that would defeat the purpose of trying to complete it as fast as possible. I made it as simple as I possibly could just so that the player can focus on maintaining their momentum.

There's not much to add in here, nor do I think any pictures are necessary. This mode had no new challenges for me to encounter, but yet somehow I still find it a fun mode to play. In future iterations I could make it less linear and have randomly generated levels, or do something similar to Mario Kart and just have pre built levels. For Project Phoebe I only required a single level, and it didn't need to be complex. Overall, despite it's simplicity I think it adds a little more to the entire game.

Survival mode

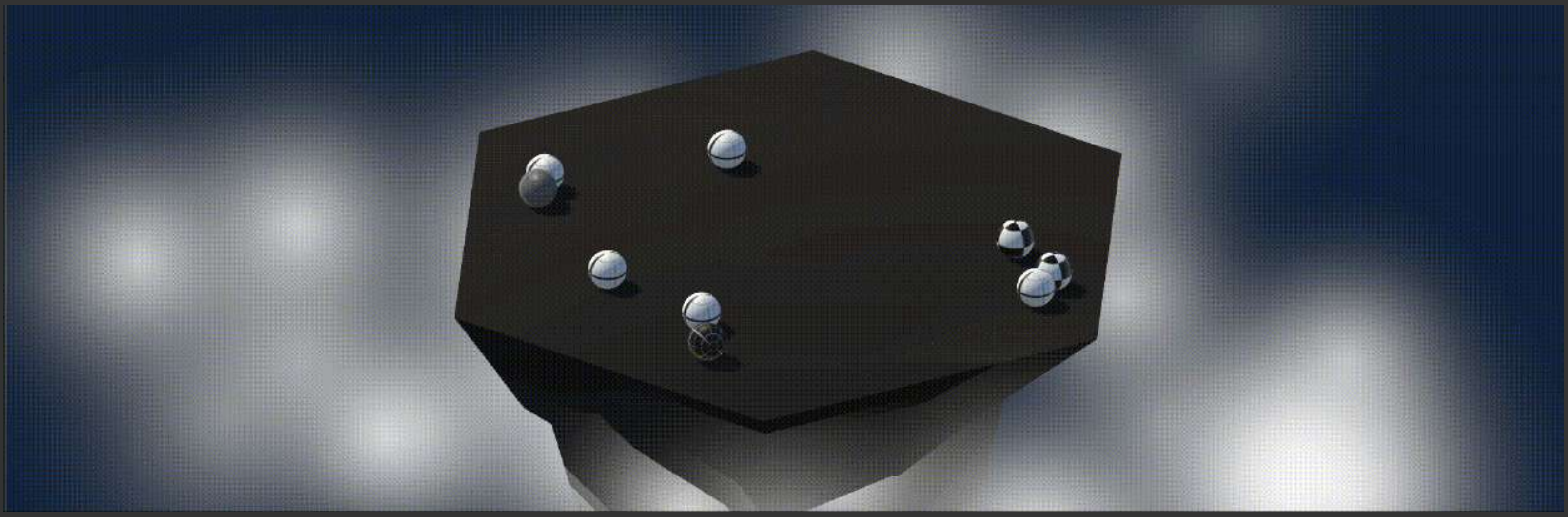
As I went further into the game, I realised that what I was creating was quite fun, outside of the general linear puzzles, what if there was a game mode where you could just battle the enemies? Inspired by "Prototype 4" I thought making a mode where it's just about surviving wave after wave of enemies. I thought having a power up to help you was similar to the goal of pressing the pressure plate, and would aid you in battle these enemies, as for the most part you'll just be running away. The power up in question gives you a stronger burst allowing you to push the enemies back, and not just get hit. I wanted the power up to spawn every other wave.

Quickly I realised that it was very easy to just horde the power ups, and even with huge waves you'd clear it easily because of how powerful the power up actually is. Even though it spawns every other wave that wasn't enough. From the idea that enemies can deactivate pressure plates, I thought I could add the same enemy in here, and have them go after all power ups on screen before they come for you. This stopped the horde but didn't make it any harder. To be fair if they can steal your power ups then they should move slower. Adding slower enemies



made it even easier, despite them stealing the power up.

To balance this, I added faster enemies that spawn in the later rounds. This quickly became harder and made me play the mode differently, actively trying to pursue the power up not only to deal with bigger waves, but to deal with these faster enemies. Inevitably once the numbers start increasing it becomes near impossible to get the power up to help you and eventually you fall off. To keep it somewhat fair, and give the player a reasonable chance, the maximum number of stealer enemies in a wave is 1, and the maximum number of speedy enemies in a wave is 2.



Final thoughts

Crazy to think how fast I was able to finish this game, and that it is actually done and playable. At this point the game is about to be put on Unity Play. I think there's a solid foundation that I can grow from with this game. Every thing I've done in this game can be done much better, and be done more elaborately. I knew this while developing the game as well, but stayed true what I was doing. No more, no less. Not only does it feel good to finally have a project built out, I know that I can make a much better game already.

The next project I make I would love to tackle the "animation" enemy. Allow myself to use assets with animations in a beneficial way. If in my next game I can make a 3rd person game with a human protagonist then the animations for that should fit the game. I'm not too worried about the look of the game, I'm more concerned about the overall theme, and I want everything I build and use to lead to the same theme.

I'm excited to make more games! I'm more excited to program systems that do something cool, not necessarily building a full game. Some of my next projects are likely to be "demos" or a really good vertical slice of what could be. I can only get better from here and I have no doubt I'll look at this project fondly later on in life, as it was the game that gave me the confidence to just build a game. I hope that if you're reading this you got the time to at least try it out, and I hope that you had fun, even if it was very short. :)