Mighty Kingdom Test 2 documentation.

24/01 – Today I received the response that my interview was successful and that I had moved onto the second step of the application process. For this, I was required to expand upon my Test 1 that I created last year. I had to however, include some additional features.

This would include such items as:

- Object Pooling (Which I am in discussions with as upon inspection of my project, I make use of a nested object pool, as well as 2 additional separate pools. One for my obstacles and one for powerups.
- Mobile Input (Allowing a mobile player to be able to press on the screen to jump and to be able to press buttons within the game to interact.)
- Scaling UI(Allowing the UI to scale based on the resolution of the display.)
- Additional Pickups.

These additional criteria should be completed by the 29th of January, so I am going to document my progress and though process behind each of these criteria.

I am also going to try and address a bug that I have found within my game to which if the player clicks on the pause button, then the player jumps as the game is being paused. Though I believe this is an easy fix, as I can just test the location of the Y coordinate and if it is less than the height of the button, do not jump.

Additional Powerup Ideas

Some of the thoughts I have in terms of new powerups include

- Slow pickup – Reduce the speed of the player by 30% indefinitely. This would be indefinite as if this was timed, there would be a catastrophic chance in speed when the timer had expired for this pickup. It would also automatically make the next speed checkpoint as the new target. This is to combat if the player would pickup this powerup, only to go past a speedup checkpoint and speed up, thus minimising the effect this powerup would have.
- Catcher pickup – Place a safety catcher underneath the player for 15 seconds. This would prevent them from falling underneath the world and having the player die. This would definitely require a visual way of telling the player how long they have on their catcher, as the catcher suddenly disappearing without a prompt is a horrible playing experience.
- Points pickup – Addition point pickups add variety to the points that the players can receive.
- Height pickup – defaults the maximum height that a platform can be to 0, essentially making all platforms level for 15 seconds.

- Pickup pickup – Increases the chance that a pickup will spawn on a platform by 25% for 15 seconds. I have thought about making this a permanent increase as well with an overall lower increase as well (such as 5%).

26/01 – This day incorporated the inclusion of the "slow down" pickup. Which, upon activated, decreases the player's speed by 25% perfect and guarantees that the player will continue at this decreased speed for 100 meters. I originally had the powerup automatically set the next speedup location to be originally what the location after that would've been, but this proved to be too easy to abuse (Stacking slow pickups was overpowered.) I also looked into implementations of mobile input as it had been something, I needed refreshing on.

I also set the anchor points of all UI elements in the game for when I would do UI resizing.

27/01 – This was an incredibly productive day, with a lot of core issues I was having with the game being fixed.

First of all, I was able to fix the bug that had been playing this project for it's entirety.

When the player would click on the pause button, it would automatically make the player jump, which to me, was a very undesired behaviour, as I only wanted the pause to occur.

In order to combat this, I made use of the EventSystem that is built within Unity and checked if the player was touching a UI element, if so, ignore any jumping based behaviour.

Another thing I did during today was ensure that the UI within the game would scale in relation to the base resolution of 1920x1080, with automatic adjustments should the resolution be smaller or larger. Because of this, I had to rescale all UI elements within the game and relocate them so that they would be larger and easier to view on a smaller display. In doing this, I also locked the screen orientation to landscape as this is the most effective way of viewing the game.

Next, I fixed a bug which was causing the player to not only stack double point powerups, but it was bugging to the point the doubled score became the normal score once the player died and reset, as well as the double points stacking regardless of if the player was picking up a no spikes pickup or a slow pickup. The way I combatted this was by setting the double points timer to 0 upon death and ensuring that the score could only be twice as high as the normal score defined.

Next, I added mobile input to the game based off of the mobile input that I used within my previous project, Xplatform. As we were required to build to a multitude of platforms. I implemented this using platform dependant compilation, which can be used in conjunction with an if statement, therefore, I am able to adjust what is happening within the Update() loop based on the platform that the game is currently being played on. While doing this however, I ran into an issue of it using both the PC input and the android input. To combat this, I made use of a simple Boolean value that will disable use of another platform input should a platform already be selected and used.

I was able to build my game to my phone and play on it, I am trying to actively figure out an issue with lag in terms of loading a scene on my mobile device, as it takes a solid 2-3 seconds for the scene to load properly and start, after this however, the game ran perfectly. I did have an issue of audio not playing when built to the phone, this though, was quickly remedied.

Tomorrow, I plan on added at least one extra powerup, as well as trying to sort out performance issues when it comes to loading my game scene on mobile.

28/01 – Today was also a very productive day, mostly in terms of clean-up of code, but I also implemented more powerups. I introduced the leveller powerup, which for the next 10 seconds, will ensure the platforms that are generated are of the same height, thus making it easier for the player to jump platforms as they are all on the same level. I also incorporated gems, which will supply the player with a large amount of points. I also linked up all the sounds for each of the pickups and added some UI to the main menu describing what each of the powerup's that you will find within the game do. I also needed to tweak the effect of the slow powerup as it was not setting the speed up checkpoint as I intended.

Once of the major issues that I encountered today though was a loss of progress due to the implementation of a powerup that I was just not able to get working correctly. I did intend on having a powerup that was sort of like a safety basket underneath the player for 10 seconds, this however caused a multitude of issues. With the first one being that the player would easily run into objects, jumps onto platforms were too high to easily get on and it was causing the strange spawning of platforms, so I ultimately decided against implementing this powerup, going with the leveller powerup instead.

I plan to do more cleaning up tomorrow before the submission, as well as doing a round of testing on my phone to ensure that controls on the phone are working correctly, I will also still need to look at possibly fixing the bug that I am facing with regards to the loading of the scene on mobile.

29/01 – There was some final tweaks made to the UI and ensured that the new font that was implemented was on all UI elements. I also slightly changed the height of the pickups to ensure that they wouldn't go above the screen boundaries. I also did playthroughs on both PC and Android to ensure that the game was functioning as expected. I did look into the slight issue on the first load up on Android for the game, but this appears to be a Unity issue.

Test 2 Postmodern

In this document, I shall be going though some of the things that went smoothly during the creation of the requirements, as well as some issues that I faced during the development.

What went right?

1. **Progress was able to be achieved rapidly… Wow!**
   I was able to incorporate all of the requirements laid out in this second exam within 3 day's and had bug fixes complete on the 4th day. This was largely based on me having already implemented object pooling within my game, as well as having the game designed in a way that allowed the game to feature mobile input. I was also able to learn the ability of UI scaling in a quick manner, and once I was able to build to Android, it made me aware of UI elements that needed to be either scaled or moved to accommodate the different phone resolutions.
2. **Unity Remote…. They have this?**
   When I last developed for a mobile platform for the game, Xplatform, we had to make use of an android emulator on the PC (which was a very clunky solution) or by repeatedly building and exporting to an android phone in order to test minor changes, which proven to be extremely inefficient. Because I was able to use Unity Remote 5 on my android phone, it was able to stream the game onto my device, making it extremely easy to test mobile aspects, such as input and UI placement.
3. **Old bugs… be gone!**
   Some of the issues that were in Test 1 featured minor bugs that, while not affecting gameplay experience, were causing behaviours within the game that were undesirable. Such as clicking the pause button would still cause the player to jump. I was able to fix this though the use of the Unity event system checking if I had clicked a gameObject(which can also be a UI element) and should the player have clicked on a UI element, please ignore any jumping code.

What went wrong?

1. **Why aren't you working, new powerup?**
   During Day 3 of development, I tried to implement a sort of "safety basket" powerup, which would stop the player from falling though the map for 10 seconds. However, upon trying to implement this, I ran into a lot of problems implementing such a powerup, such as platform's spawning way too far apart and clipping planes making it difficult for the player to see the platform, as well as the player running into the platforms at times as being stuck, causing them to fall to their death as soon as the powerup had run out. I ultimately decided that I had to implement a different powerup, therefore losing time.

Lessons learned?

- Resources are your friend. Including old projects.
- Try to better manage time and know when to implement something different when an idea fails.
- Prefabs are infact, your friend.
- Comment your code, detailing what you are about to write, before writing it. It makes **WAY** more sense that way.