Pachinko Balls Catching Game Awesome 3K Graphics, Not yet at 4K

Picking up where the counter game left us, I’ve decided to implement some features that will turn this “non-game” (no interactions) into a somewhat fun and entertaining game. To do so I’ve decided to add the following features:

* Moving the box to try to catch all the balls;
* Spawning balls on top of the map so that the player may catch them;
* Creating a timer;
* Creating difficulties;
* Putting amazing sounds.

To continue this adventure, I shall explain each of these steps in a programmer’s vision as well as a Game Designer’s one.

**Moving the box**

To move the box, I simply create a “PlayerController” script that will allow me to move the box sideways, from left to right, making the player able to try and catch every ball. In the script I simply create a reference to Unity’s Input system as well as the Axis “Horizontal”. Tie them together to a transform.Translate movement and you get yourself a moving box. To add a bit of a spicier movement, I’ve added a variable speed that will multiply with the Axis. I’ve also created constraints, that check if the player is in x coordinates and make it impossible for them to continue further, so stopping the player from moving off the screen.

A screenshot of a computer program

Description automatically generated Image 1 : PlayerController Script

**Spawning Balls**

Okey, so spawning balls seems simple, doesn’t it? It was. It was very simple, not hard at all, yet I’m a tryhard and I wanted to do better, and so I did. For starters I imported the ObjectPool script that the pathway gave and started analyzing it a bit, for I wanted to use the SetActive feature instead of the Destroy one, it’s better for optimizing, and even though this game probably won’t be potent in that aspect, I did it, just because. I attached the ObjectPool script to the SpawnManager object I created and then created a SpawnManager script. I also made the sphere a prefab, and attached a script that would de-activate the ball when it touched anything. For the SpawnManager script I created two variables, the initialSpawn that states how long it takes for the first ball to appear and then the spawnRate. Then I created the SpawnBalls function that would take the GetPooledObject function from ObjectPooler and attribute it to a temporary variable called ball. Then creating a variable called spawnPos, I made the Y (height) 19, outside of view, and the z, a random number between the minimum position of the player and the maximum position it could reach. Setting the ball to true and making it’s position the spawnPos would create the falling balls effect.

In terms of design, its cute, random balls falling is the mechanic and the player moving around trying to catch them is the dynamic. Yuupi balls falling, here’s code, have fun.

A computer screen shot of a program

Description automatically generated A computer screen shot of a program

Description automatically generated

Image 2 : ObjectPooler Script Image 3: SpawnManager Script

**Creating a Timer and Creating Difficulties**

So, this is a long one… Creating a timer and difficulties required me to establish a game over, a game start, the buttons, the events, I realized some code needed to be reworked and so here I am, practically with the game over, having only sounds to be inserted. Making the timer, easy peas, I finally made the Game Manager Script and object, then I realized, wait why do I need the SpawnManager? And so I deleted it, and put the code on the Game Manager, making it basically the Spawn Manager but in a spicier way. Doing that, I created UI for the timer, simple, basically just copied the Score UI. With a Coroutine I check if the game is playing (I created a bool variable called gameIsPlaying) and then I wait one second, take 1 from the time and change the UI, in case the time was zero, the game over function would play. This is the countdown, as simple as that, yet I realized, I need to make the GameOver function, and so I did. Pretty simple, like the other Junior Programmer Pathway one we did in the previous lesson, I create a GameOver TextMeshPro, and put the game over text and a restart button that calls the RestartGame function in the GameManager script that restarts the scene. In the game over text, I also put the score the player had gotten, requiring me to make a reference to the Counter script in the box. This helps a lot going into creating difficulties, and so I did. Importing a font to make the title text prettier, I then created a DifficultyButton script and gave it to each of the buttons. This script contains a difficulty variable that changes according to the button, 1 for easy, 2 for medium and 3 for hard. Each of these would then pick the spawnRate and divide it by that value, getting the difficulty established. I then did some clean up, some comments to explain the code further. In the TouchGround script, I added a check to see if the ball touched the player, so that it wouldn’t take time when hitting him, I also deactivate the ball when it touches the trigger.

In terms of design, the time creates a sense of risk for the player, yet a new sense of reward as well, as when the player gets a ball, the timer moves up, creating a figure of control he didn’t have before. I also made it so that when the difficulty is 3, instead of just spawning the balls faster, a second wave of balls would also spawn, making the game even harder. The game over screen displaying the score is also a good reminder and challenge for the player to try harder. When the ball hits the ground, time will be taken off so then incentivizing the player to keep moving and catch every ball possible. So here’s the last bit of code, before the final result.

A computer screen shot of a program

Description automatically generated A screenshot of a computer program

Description automatically generated A computer screen with many colorful text

Description automatically generated with medium confidence

Image 4, 5 and 6 : GameManager Script

A computer screen shot of a program code

Description automatically generated

Image 7 : DifficultyButton Script

**Amazing Sounds**

Sounds, got them from this site: <https://pixabay.com/sound-effects/search/game/?pagi=4>, they’re put to good use. We have music when the game starts, a sound for when he clicks a button, when he catches a ball and when he lets a ball fall, there’s also one for when the game ends. That’s it, that’s amazing sounds, they are amazing indeed, 8 bit vibes, classic video games.

**Conclusion**

Well, with this I think I did all I aspired to do, got all the features I wanted in the game. I changed a little bit, improved some coding, added comments, and made it so barely just the touch of the box would give the player the point. This fixed the bug where some of the ball’s rotations would go crazy, since they touched the object and then rotated, yet touched the trigger inside too, so I just made the whole box a trigger. Another thing I changed was the second wave spawning in the highest difficulty, I added some time to the initial delay because it was spawning at the same time as the other ball.

And that’s it, that’s Pachinko Balls Catching Game Awesome 3K Graphics, Not yet at 4K. An amazing game, made in one day, that actually made me even more inspired to continue programming and developing games, hope you enjoy it as much as I did making it!

Link to code: https://github.com/ManelNeon/Counting-Prototype

Link to play: https://play.unity.com/mg/other/webgl-builds-394088