Brick

Introduction to Games Programming

Shaun McCusker

S2003216

Computer Games (Design)

*I confirm that the code contained in this file (other than that provided or authorised) is all my own work and has not been submitted elsewhere in fulfilment of this or any other award*.

* Shaun McCusker

**Project Description**

This is the documentation of Brick, my Breakout-style game. I made the game using C# in Unity, using mostly Unity built in Assets, with the exception of the Brick sprites and the music/sound effects which I made myself. It has been fully coded by me, taking into account player controls, collision, particle effects and audio. This was my first major project I worked on solo.

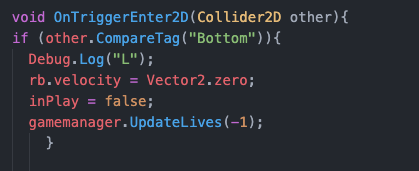
**Code Description**

There are 4 main scripts written for Brick: ball, brick, paddle, and GameManager.

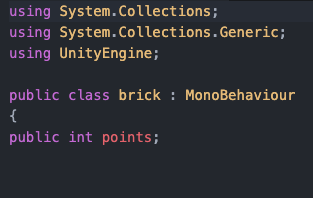
*Ball*

This script defines all the attributes assigned to the ball, including what happens when it collides with anything or falls off the bottom of the screen.

This section in particular shows everything that should happen when the ball collides with anything with the tag “brick”. It applies the explosion particle effect, destroys the block, updates your points (different brick colours carry different point values), updates the total number of bricks so the game knows when to end, and plays the audio file assigned to it.



This code displays what happens when the ball falls off the bottom of the screen. If it makes contact with the collider tagged “Bottom” it will reset the ball to be above the paddle, and remove one life from your total. This total is 5 at the beginning.

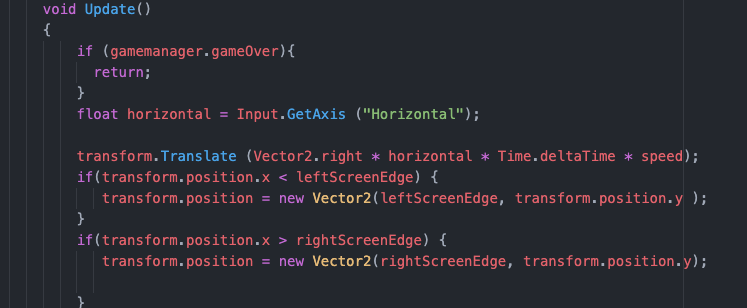


*Brick*

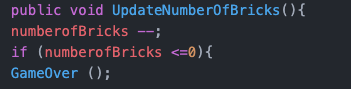
The actions of the bricks are all taken into account by the ball script and the GameManager script, so its merely there to be able to correlate with the points system in-game. The green bricks are worth 100 points, blue are worth 250, pink are worth 500 and purple are worth 1000.

*Paddle*

The paddle is controlled by either the arrow keys or A and D, as is setup with this code here:

This code also stops the paddle from ,being able to go off the left or right side of the screen, with it setup to collide with leftScreenEdge and rightScreenEdge, which are collision boxes. In unity I also angled these boxes inward slightly as I found that if the ball is travelling mostly horizontally, it can get stuck or just take ages to return back to the paddle. Angling the boxes inwards fixed this, as it forces the ball to always have a bit of downwards momentum applied.

*GameManager*

The GameManager script is responsible for all other functions of the game. This includes The updating lives and score text on screen, how the game knows to game over when out of lives or all the bricks are gone, and making the game over screen functional. The code here is for the updating lives and score counters onscreen. It is setup to work with the Canvas in Unity.

The above code is responsible for the game knowing when you have completed, and there are no bricks left. It is an active counter of how many objects onscreen are tagged “brick”, and when it reaches 0 it will call on the game over screen.

**Challenges**

I found the mechanics of the ball breaking the brick to be quite challenging. I knew it was essential to the game, but had never used the Instantiate function before, so had to do quite a big of testing to see how to implement it in order to get the desired effect. I had never fully coded my own project from scratch before, so was quite intimidating knowing just where to start with it all. There was a lot of trial and error involved with all aspects of the scripts, but I feel a lot more confident with all of it now. I also had little experience with adding music and sound effects to my game, but that turned out to be quite simple once I knew what I was doing.