Inkball Report

540739815

Design Choices

In my program I decided not to use any inheritance or interfaces, as I decided that there were no objects that shared enough properties/functionality to justify it. I considered using it for the walls/spawners/holes, and having them inherit from a 'Cell' class, but I opted to use just a Cell class for all of them, with a 'type' attribute to differentiate them. I chose this because there were little benefits to splitting these classes into separate classes. While they do share many attributes and methods, I decided they didn't differ enough to justify the clutter of their seperate classes, as the only differing functionality was that walls collided with balls and holes captured them. Another reason was that I planned to implement the cracking bricks extension task, which requires the type of cell to be changed from a solid, collidable wall to none when it is broken. With my design structure this can be done from within the object itself, whereas if each type of cell was split into its own class the responsibility of this task would have had to been delegated up to the Level object which contained the cell. Other than Cell's there were no real considerations to be made surrounding an implementation that made use of object oriented features past classes, as the app, levels and balls were all completely separate.

Implementation of Extension Feature

I chose to implement the breaking bricks feature. This required all walls to visually 'crack' as they were hit, and completely break after being hit three times. Furthermore, colored bricks could only be broken by a ball of the same color. In order to implement this I used my already existing collision handling method, as it was a part of the Cell class and I could therefore change the wall's status from inside the method. I kept track of the number of times the wall had been hit with a counter variable that was incremented when a ball of matching color hit the wall, or anytime a grey wall was hit. When the wall was first hit I changed its sprite to the cracked wall with its corresponding color, and when it was hit three times I changed the walls type to 'none' (meaning not a wall), and the sprite to the background tile. By changing the type to 'none' the Cell would no longer collide with any balls.