Bryce Daniel

December 12 2014

Final Game - Nodes

Part 0 - Preface

Before we get the essay, I would just like to say it is a great thing that you granted me an extension because about a week or so ago I decided to scrap the entire project and come up with this one. Overall it was a good learning experience, and I ended up being able to refactor some of the code. Ultimately this is a better game though and I am happy with the way it turned out.

Part 1 - A game and its Rules.

Like game one, a good game has to follow a set of specific rules are laws that govern its play. If these rules were to cease in their existence, there would be no game either. In Nodes for example, the nodes must match the color of the ring to score. This means we cannot just score anything that waltzes into the ring. Oh no, we have to check the colors, like passports at a border. The Nodes instead also must be created in the bounds of the screen, but not where the ring is. This is just one example, but we can already see how important rules are to the game. To test the game, we must look at the game model. Most tests can be checked by letting the game run on its own but a few need some interaction, like checking button presses do what they ought to do.

Part 2 - The Model

The game Nodes follows the rules as laid out in The Manual. In order to represent the rules of the game into rules that the computer/iPhone can understand, the model is implemented in Xcode using the Swift language and SpriteKit API. Briefly, let me touch on SpriteKit and what I found to be its benefits as well as its shortcomings. SpriteKit allows me to create SKNodes which are essentially arrays of SKSpriteNodes. For instance, I have created the game scene which is a SKNode. Everything seen in the game from the text to the circular nodes and ring in the center are then children of this node. With built in functions such as “addChild()” and “removeFromParent” I can control what is displayed on the screen at which time. This makes it fairly straightforward in terms of managing graphics and updating the display. In game one I thought that I could only access the first and the last in the array but after experimenting I found that I could use .children[index] within a for loop to iterate through the entire array! This made changing the color of the nodes and other tests possible.

The model of Nodes, without going into incredible detail is as follows:
The GameScene class is where the majority of the code lives. There is some code in other files, but those are generated by Apple and not really of any importance to us. The Nodes code that we care about can be found in GameScene.

The scene was setup by changing the background color, adding gravity simulation and creating a PhysicsWorld which will handle all of the collision simulations etc.
The function calls on line 100-106 like setupBackground() setupScoreLabel(), setupHighScoreLabel(), setupCenterRing(), spawnNodes(), spawnPowerups(), setupPauseButton() call their respective functions that create the object, adding them to the world. It defines their size properties and locations relative to the screen width and height. It also defines them as physics bodies so they can interact with each other. Gravity has been disabled so that they do not fall to the bottom of the screen. setupNodes() on line 508 is a bit more complex. It sets up the size of the nodes as well, but it also generates a random x and y position on the screen. The node is then generated within specific parameters to determine that it appears on screen but not in the center ring It also handles generating a random color and initial velocity in a random direction. The nodes act as the mobs but also as user controllable elements. [More on this later]. spawnNodes() on line 639 does the rest of the work, generating the nodes with a delay and repeating that action forever. The same is done for the powerups, but a the delay makes sure they less frequently. Similarly there are functions for setting up the score and the high score labels, he replay button at the end and the pause button at the top. The high score label works by checking to see if the score is greater than high score. If the score is greater, the high score increments as well. The timer at the top is what determines when the game is over and it uses a delay function to detriment the time variable every 1 second. It is constant unless you get the freeze clock powerup.

But how does the model keep score? Well I'm so glad you asked. On line 746 updateLives() does the work. It checks first to see if the node is within the area of the ring in the center. If it is, between the min and max x and min and max y positions, it will then check to see what color the node is and compare it to the ring. If they match, it increments the score and the frenzyBonus (when you get ten in a row, you switch game modes) [More on this later]. It then animates the node out of the scene and checks to update the high score. If the node's color does not match, it detriments a point, resets the frenzyBonus to zero, and animates it, this time fading to black and shrinking. By chaining the name property of the node once it has been scored, it insures that it cannot be scored more than once while it is animating off.

Part 3 - The Mobs

Like I said previously, the nodes are both the mobs and user controllable “nuisance characters”. When they spawn they have a random color and a random trajectory. They launch off in any way. But anywhere you tap on the screen they move to. This is a blessing and a curse. You can get them to go where you want but they move as swarm (or mob - pun intended). However, you can drag just one node to fling it into the ring. But the annoying thing is, they will still follow based on the one you are controlling is. Its a mob mentality, they cannot help themselves.

Part 4 - Frenzy Mode

There are two modes in this game. In regular mode the nodes come every 1.5 seconds. It is a nice fun pace that gives you something to work with, without being boring or overwhelming. Now if you happen to score 10 in a row without scoring the wrong color, you enter Frenzy Mode. The screen darkens (as seen in the manual) and the nodes spawn at a faster rate of 0.2 seconds. There are a lot more nodes on the screen now, but luckily, in Frenzy Mode, you cannot lose points for putting the wrong color in the ring. Its all colorful goodness. Frenzy lasts for 10 seconds and then everything goes back to normal. All of the frenzy modes that you did not score disappear, and the screen is white once more.

Part 5 - Persistent Player Attributes

There are two player attributes (besides score) in the form of two powerups. They spawn like regular nodes, but when you tap them, their indicators light up and you can tap them to use them at your will. The first, MonoMode changes all the nodes on the screen to the same color so you can rack up some quick points and even go directly into FrenzyMode! Talk about strategy. The second is a FreezeTime clock which stops the clock, but not the action. It basically gives you 5 fore seconds of node capturing fun! Once you use them, the indicators return to normal.

Part 6 - Testing the model.

In the background, the game runs a series of tests before each frame is rendered to make sure that the game is following the rules we have stipulated. The outputs are then streamed through the output console and a shouldBeTrue boolean is monitored. Let us now examine the specific tests that are run.

testThatHighScoreisMax() on 145

This test checks that the high score is the highest your score was during a play session. It checks that the high score is always greater than or equal to the score integer value. It should never be the case that is less, and if it is, there is an issue with the game model. It also shows however that the high score does not decrease with a node of the wrong color either because it is meant to be the highest amount of points your reached during the round.

testThatGameEndsOnTimeUp() on 161

When the timer reaches zero, the game should be over, because if not, then the clock has no real purpose and we are not following our model. However, the game should not be over before the timer runs out, so by using a series of if statements that checks for the “gameOver” boolean to be true and the timer Int to be 0, we are able to test that our game’s countdown clock is working effectively.

testThatNodesGeneratesWithinFrame() on 179

A node should never spawn outside of the fame of view. If this happened, there would be a lost node wandering around in the world, and we cannot have that. Furthermore, the game would be no fun if there were no nodes being generated in the center ring. This code tests for the conditions that the node when it spawns, has a position greater than the minimums x,y and less than the maximums x,y, but also not within the frame of the ring, to ensure that the randomly generated nodes pop up in the right location.

testThatNodesMatchRing() on 197

It is probably most crucial for the game play that you only get a point when the ring color matches the color of the node you are trying to score. If they do not match, but you are still getting a point, then our game model has a serious flaw. Furthermore, if you are not getting points even though the colors match, then something is wrong and there will be riots in the streets! Luckily for us, the model works just fine and we get points only when the colors match.

testThatTenIsFrenzy() on 226

Part of the assignment is the ability to switch between game modes. Well this function tests that we are able to do so. When you consecutively score 10 nodes of the right color in a row, you should enter FrenzyMode. The test checks that only once the FrenzyBonus integer reaches 10 does the game switch over to FrenzyMode. It also checks that FrenzyMode is not occurring while the FrenzyBonus is not at 10, because that would just be too much Frenzy and not enough following the game model.

testThatMonoChangesColor() on 244

The last test checks that the power MonoMode does what it is supposed to. When you press it, the nodes .color value should all be 1) the same and 2) the same as the ring color. Using a series of if statements we are able to determine that this does happen when the powerup is used, but only then. Not just all willy nilly because that would make the game to easily won if they are always the same color.

From the above tests, as well as playing the game, we can see that the game model works as desired. Everything is set up and runs as it should. The nodes spawn random locations within the fame, the points only increase when the correct color node is scored, and the game enters frenzy mode at the right time. Nodes is a fully functional game ready for the App Store (in fact this one I might upload sometime in the near future) and with this essay to prove that the model is working, along with the fun gameplay video, who wouldn't want a chance to play it!

For one physics simulation round, the output was:

```
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
```

```
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
```

```
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
Pass: High score is greater or equal to current score
Pass: Time is not up and game is not over
Pass: Node is generated within the frame
Pass: FrenzyBonus is less than 10 and frenzy mode not activated.
[ Many more pages of the same ]
```