Emily Moeller

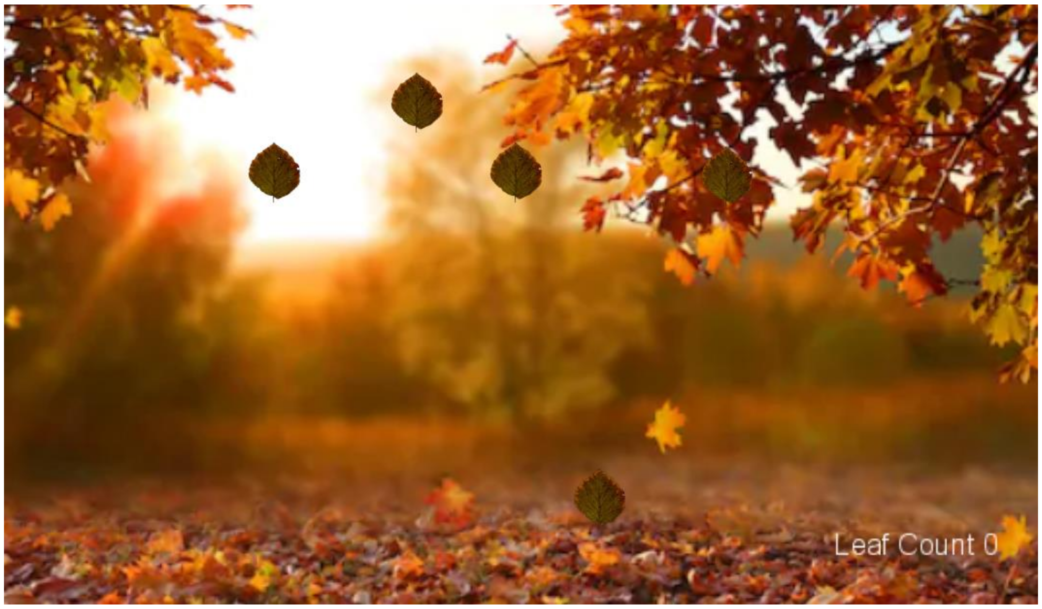
9/23/2019

Homework 1: Balloon Tap game

Gameplay/Game Design:

-Give a brief summary of the game (along with a screenshot). What's the goal? Are there any points? How are they scored? Any powerups or other special abilities?

The object of the game is to collect (i.e. click) 50 leaves that are falling from the trees. Each leaf corresponds to one point, so after you hit 20 points you have won the game. There are 5 levels, therefore when you collect 10 leaves you advance to the next level. As you progress through each level, the game becomes more difficult in a few different ways. In the first level, there are many leaves that fall slow and in a straight line. In the second level, some leaves fall a bit quicker and one slow zigzagging is introduced. A slow, straight falling acorn is also added. If an acorn is clicked, three points are deducted from the total leaf count. In the third level, there are less straight falling leaves and more zigzagging leaves and straight falling acorns and each of the leaves falls a bit quicker. The fourth level becomes more complicated by having mostly zigzagging leaves and straight falling acorns. Lastly, the final level has zigzagging leaves and zigzagging acorns. Most leaves and acorns have random starting locations throughout all the levels, so that the game does not become too predictable. The last couple of levels also have less leaves, so it becomes very easy to click an acorn.



Beginning of the game.

-From adesign perspective, what are some features you didn't get to? How would you improve your gameplay if given more time?

One feature that I didn't have time for was implementing more complicated, smooth, curved movement functions for the leaves. If given more time, I would have used functions like lerp to make this movement possible. To make the leaf movement more unpredictable, it would have also been nice to change the pace of the falling leaves mid fall, instead of when they are regenerated at the top of the screen.

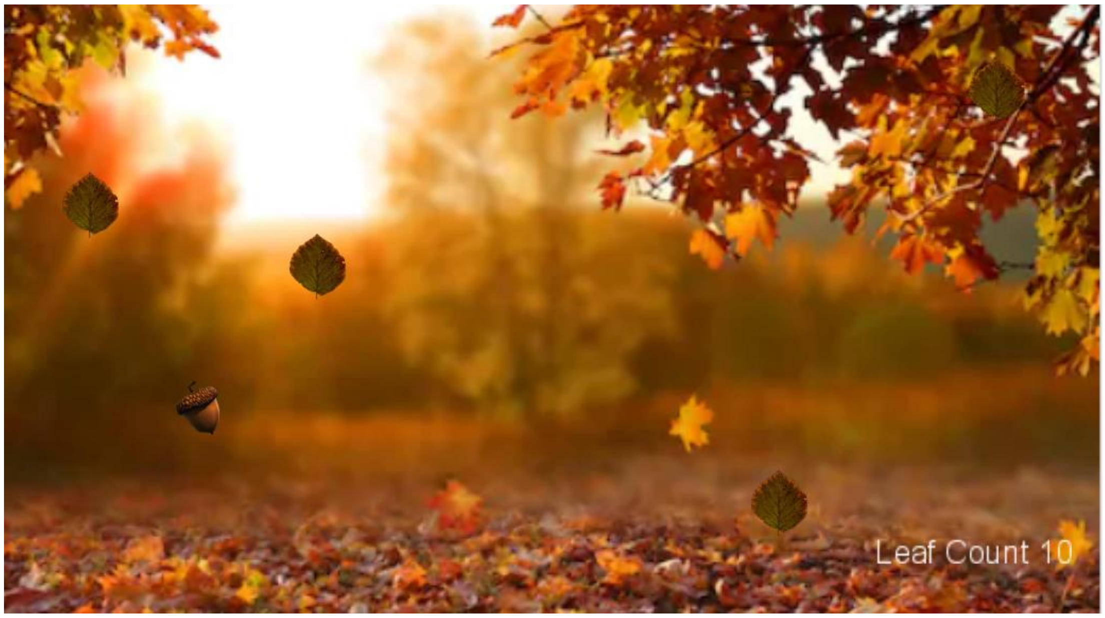
I would have also liked to make the game more playable without outside explanation. For example, sound effects or special graphics would help indicate that acorns are not supposed to be clicked. A graphic could have been added in to indicate a squirrel attack, which would help explain the loss of points. Currently, the total leaf count just drops after the acorns is picked, which could be confusing if you had never played the game. In addition, there is no indication that you have advanced to the next level, besides a clear increase in the difficulty by how the leaves and acorns fall, so it would have been nice to add in a temporary graphic or sound effect to inform the player of this.

From a purely aesthetic perspective, I would have liked to add in different kinds of leaf objects, potentially having different colors of leaves correspond to different point levels and difficultly of catch. I would have also liked to add in specific events, like a gust of wind that would trigger the leaves to fall in an even more unpredictable fashion.

Algorithmic Discussion:

-**What are the key algorithms and techniques you used in developing your game? How do these algorithms improve your computational runtime or user experience?**

One key technique for this program was creating a leaf object with various variables that dictated how the object would move. It was important to be able to reuse code and make it easy to modify the behavior of falling leaves. For example, I could individually track which leaves had been clicked, so when one leaf was chosen it didn't impact the movement of the other leaves. I also was able to make the leaf move unpredictable by using random variables to control the starting location and the speed of the falling leaves, which provides a more interesting user experience. The Leaf object also ended up being used for the acorn objects, since they are both images that have similar movement. In addition, loading the images and instantiating objects for the leaves and the acorns in the setup method helps improve performance.



A new level starts at 10 leaves, which is when the first acorn appears.

In addition, I also have only one method that controls the translation of the grid, which in turn is where all the leaves are drawn. This made things simpler for me to understand and debug, because I knew that the grid could only make one translation and then it had to be set back to it's starting place. Therefore, all the movement of the leaves is calculated in other individual methods, but I was able to reuse the same method for the actual drawing.

-From a computational perspective, what are the bottlenecks/limiting factors in your game? If given more time, what algorithmic improvements would you make?

Current all the leaves are individually declared at the top of the program, but in an ideal world, Leaf objects would have been generated in random quantities. Creating more generalized movement functions that are largely dictated by random values, but still result in smooth movement, would have also been preferred.

Also, the structure of code in the movement methods, like zig zag and generic straight fall, is similar so that code could have been refactored to be more concise and clearer. It would have also been nice to have the movement of the leaves become increasing more complex and quicker, without hard coding those values, but instead have an automated gradual increase that would allow for more levels.

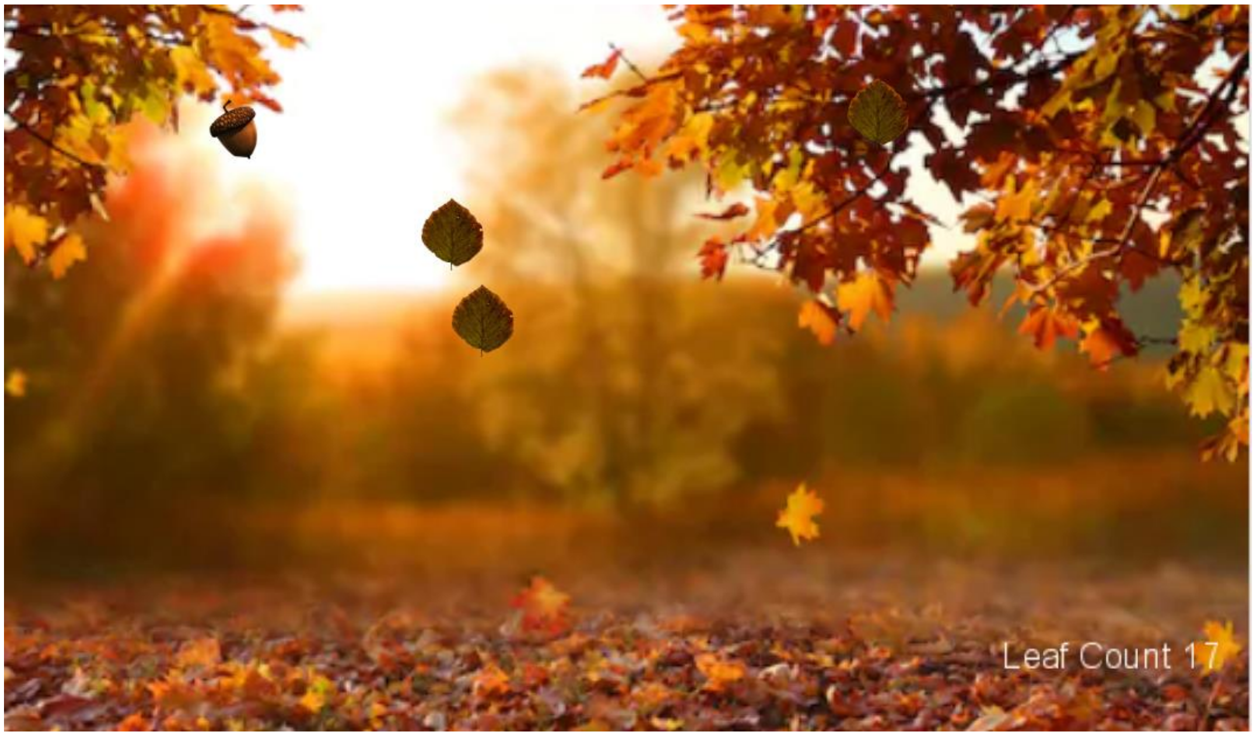
Game Engine Analysis:

-Processing provides fairly low-level support for game development. What aspects would have made your game easier if they were already provided? Which of these would be useful for other games?

It would have been helpful if there were more simple ways to create unpredictable, curved movement of an image object. It also would have been nice to have prefabs to use inheritance to design the falling objects in a more organized way. Prefabs also useful for other games, because reusing objects is very common and it is also helpful to have variations on those objects, which is where inheritance comes into play.

-In the process of creating your game, you've already made some key aspects of a game engine. Describe the key components of how your game works, and how they relate to each other. For example: what systems are responsible for triggering sounds being played? Where is the state of the game stored?

The state of the game is being tracked in the draw function via the leafCount variable, because the entire game revolves around how many leaves have been collected. The leafCount refers to what level you are on, which results in a different state of leaves that fall. All the code for the levels is contained in the draw function, which is called continuously, so once you have clicked the appropriate number of leaves you immediately advance to the next level. Each leaf also has a variety of variables that dictate its own state such as speed, starting location, type and clicked. If a leaf is clicked, it won't be drawn until that leaf would have finished falling to the bottom of the screen.



The Leaf Count variable is shown in the bottom right corner.