Iskandar Shoyusupov & Emmet Easter

Project 6

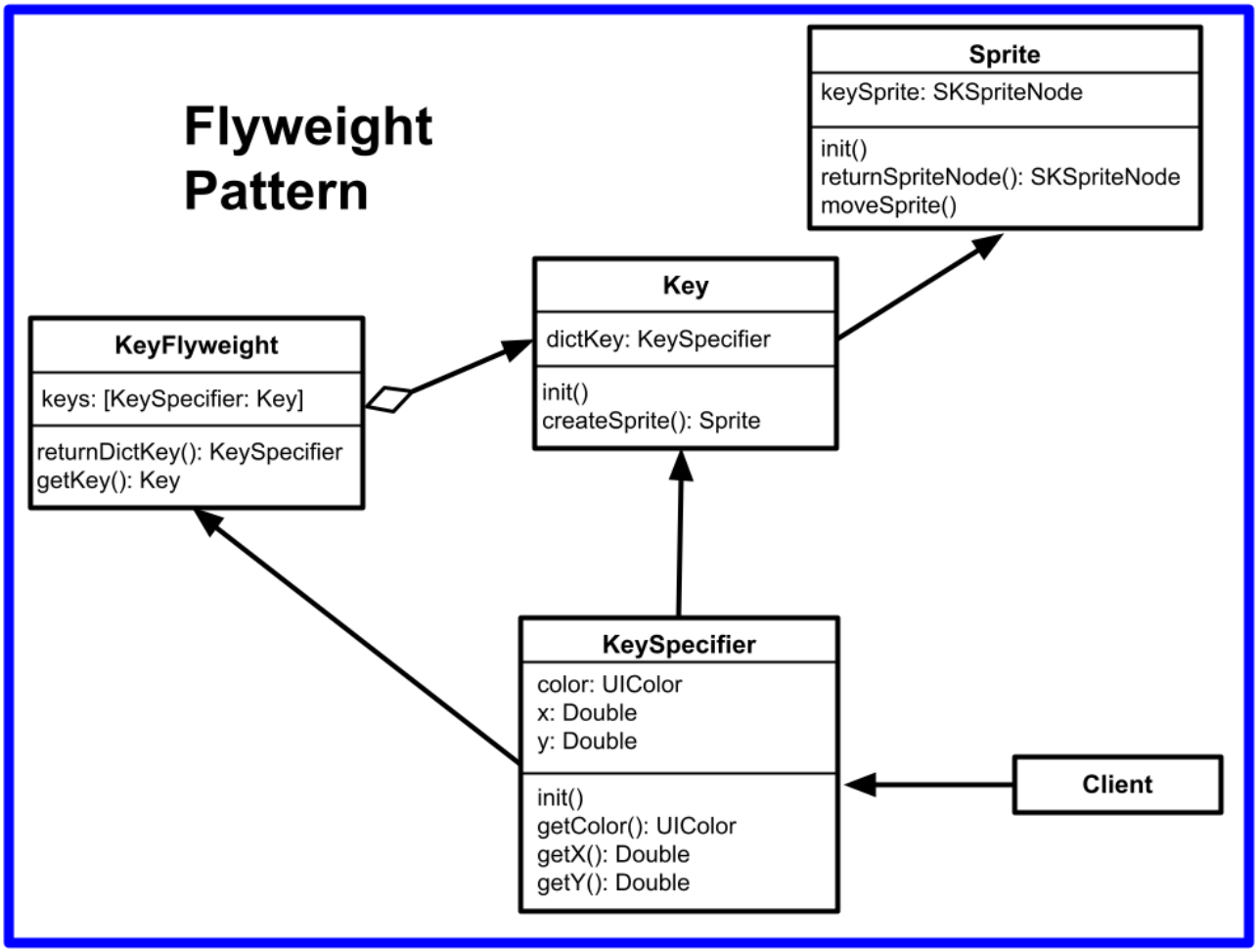
Status Summary

• Work Done:
- Emmett: Swift/ app implementations
- Iskandar: Setting up audio system on Raspberry PI and proper TCP connection server for communication between the Swift app and the PI

• Changes or Issues Encountered:
- When creating the tiles, I learned that swift doesn't have a built in library that allows objects to move. So I had to use the SpriteKit library, and adjust how some functions worked.
- Planned to use a HW-125 module to store WAV files for each key tone, and then send them through a serial write to a two transistor amplifier circuit and then to an analog speaker, however there were many issues encountered trying to implement this. Unlike Arduinos, Raspberry Pi's don't have an inbuilt ADC (analog to digital converter) meaning that all audio would have to be transmitted through PWM (pulse width modulated) signals. This would be fairly complex since WAV files have header formats that aren't easy to read and specific frequencies would have to be set.
    - Change: Instead, audio output will be sent to a built in audio jack through the python pygame library. Specific tone sounds will be generated through a musical beeps library that is able to play individual tones across all octaves
    - Changed from UDP protocol to TCP, simpler to implement
- TCP Server sends data either too slowly or the musical beeps library is not well implemented. Still trying to figure out how to serial read WAV files to maybe implement a better solution

• Patterns:
- So far, implementing a flyweight factory pattern has been used to create the piano tiles sprites. This helps, since the only difference between the keys are color and x coordinates. It's much easier to create tiles, and is easier to read.

Class Diagram



Plan for Next Iteration

- Implement levels
- Implement score
- Implement reading Raspberry PI input
- Implement comparing input
- Fix note playing to be more smooth
- Implement observer pattern for localhost server
-