Evan Shearer & Seth Rapp

CS2021

April 2017

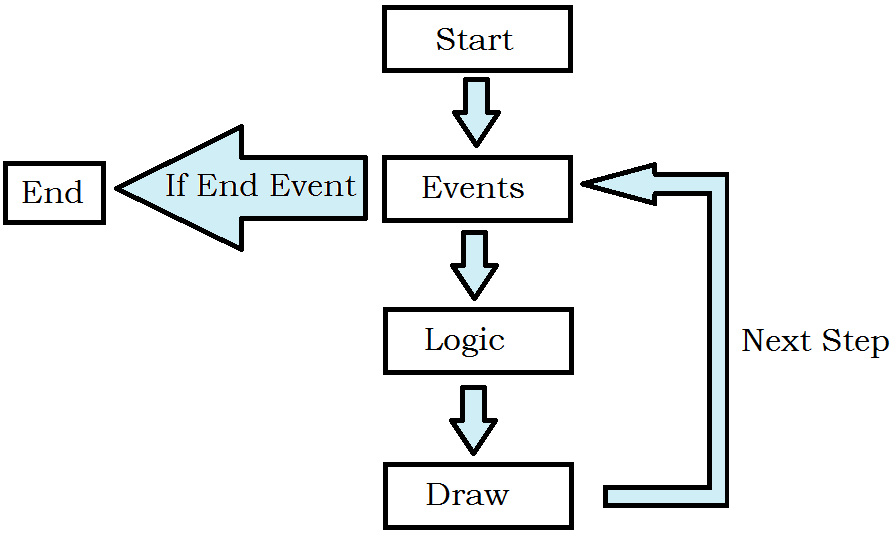
Final Project Report

Introduction

The project goal was to use an external code library for python to create a presentable project. The Library we chose to use was Pygame. Pygame is a library with classes that are meant to be used to code computer games. The goal we set for ourselves was to use Pygame to create a simple game that is playable. We chose that instead of writing code that meets its specific purpose, we chose to write code that could easily be rewritten for future works, or at least easy to change later.

When choosing the structure of our game we considered what Pygame can do, in addition to what would be easy to model beforehand. We chose to make a 2D platformer. We considered doing a top down platformer, but doing so would require more moving parts for the same complexity of game, we did not want to get bogged down by a lot of objects interacting, so we went with the 2D platformer option.

The game runs off three types of files: the main file(ensgame.py), the game file(game.py), and the sprite files. The main file creates and starts an instance of Pygame while declaring the clock and screen objects. Then the main creates an instance of the game and steps through the three logic steps at 60 ticks per second. The three logic steps that get run from the main are the handle\_events, logic, and draw\_frame.



**Game Flow Diagram**

The steps exist in the game file as functions. The handle\_events gets user input and makes decisions based on what buttons are pressed. The logic step allows for decisions to be made based on the current game state. Lastly, the draw\_frame step allows for all the current objects stored in the game to be drawn to the screen that the user sees.

The final part of the project is the sprite files. The sprite files include objects such as blocks and the player. The sprite files are children of the Sprite class that is a part of Pygame’s library. They contain data such as their position and image. The sprite children also have added information based on what actions they need to do and logic that is needed to be performed. These files and objects together form the game we set out to make.

Project Results

The original goal was to make a 2D platformer that played as a Metroidvania. A Metroidvania is a game style that typically involves progressing through a system of room and having to backtrack as new powers or abilities are obtained. The main issue with trying this would not come from level design, but the fact that a Metroidvania would need to incorporate a lot of new mechanics and abilities. Due to the long time it took to make a workable environment with a controllable player, many defining aspects of Metroidvanias were not implemented into out game.

Our game still succeeded on many aspects we wanted to have. The environment consists of rooms that the player can travel between and backtrack through. With the inclusion of splitting paths, this system of rooms slightly mirrors the style we were aiming for. Another pothole in our design was the fact that neither of us were artistic. With no artistic ability between us, we were left with very dull and poorly made sprites.

Looking at the goal to “use Pygame to create a playable game”, we succeeded. The game runs from opening ensgame.py with all other files in the same directory. Python and Pygame are needed to also be installed, but this is easily done with a machine with Python3 and running ‘pip install Pygame’. A python program exporter could be used to get it to run without such, but was not needed.

Since there were many resources online to learn the Pygame library and common practices, the hardest issue was the time constraint. The time given to finish the game was plenty, but not enough to create a full game with many features. The default viewing windows resolution and windowned state was a decision made to focus time on designing the game.

Division of Group Work

There were only two members, Evan and Seth, working on this project; because of this, work mitigation and sharing of responsibilities was easy. Evan started by researching Pygame and finding useful resources to self-teach ourselves Pygame. We collaborated, discussing game design and possibilities for the game’s type. Seth, in the end, decided that a 2D Metroidvania-esc game would be better in terms of quality that we could produce over a top-down game.

Evan Started the code by making the ensgame.py file and creating the game.py file. We both worked together to sort out the sprite’s children classes and determined how the player exists and interacts with other objects. Evan wrote majority of the code for the block and player classes while Seth wrote all the room classes, including their parent class. In the end, all the code was finished up by them both.

Self-Assessment Paragraphs:

By Evan:

The project was a very fun thing to work on. My contributions were many but could not have made it all without Seth. I ended up making the platform for the game to run on (ensgame.py and game.py). I made the simple item class that was a child of Sprites so that the player and block classes could be used consistently. I also made the player class and the block classes for the most part. I had also made color.py which was meant for easy RGB color usage, but was almost never used.

By Seth:

Bibliography

Code Appendix