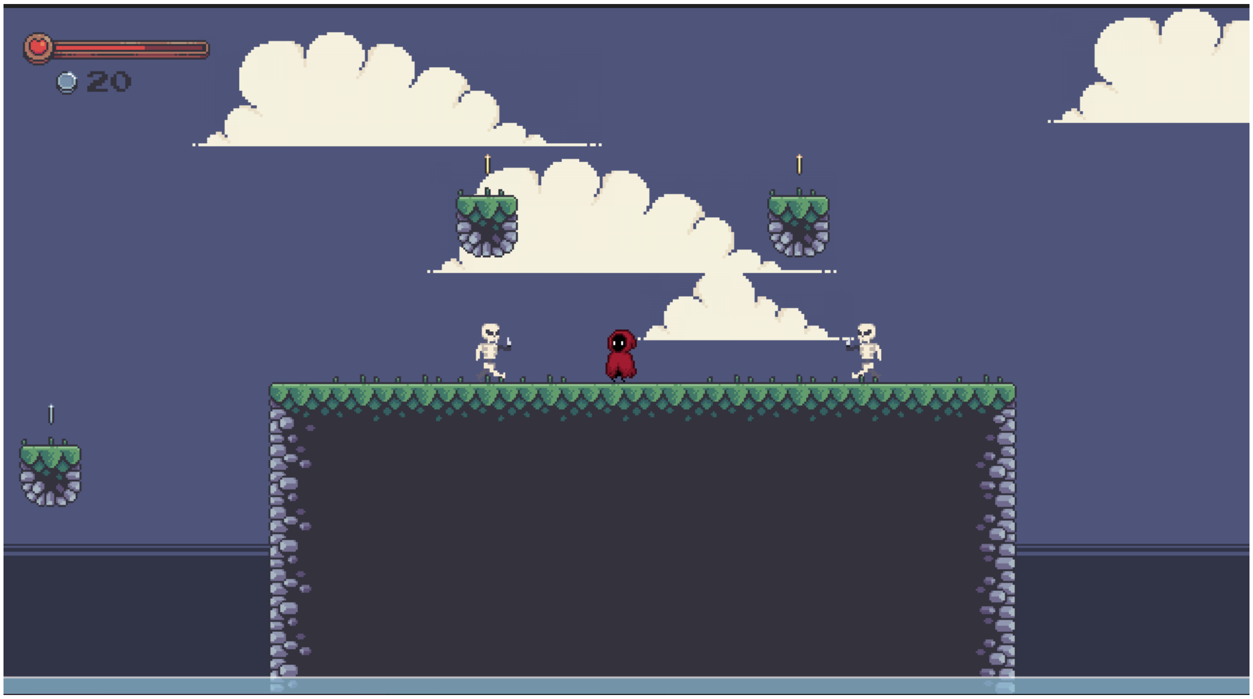
Game Document

Michael Elrod

Part 1: Introduction

- The name of the game is Mystic Meadows
- https://www.youtube.com/watch?v=_754X8bVQW0
- To play the game: download the full repository from Github and run this command:
  python3 Game.py
  from the terminal with the following installed:
  Python Version: Python 3.10.6
  Pygame Version: Pygame 2.1.2
- Controls:
  - Movement: Left & Right arrow keys
  - Jump: Spacebar
- To play the game, traverse the platforms to reach the goal at the end and move on to the next level without dying from either running into enemies or falling off the map. To kill enemies, you must jump on their heads.



Part 2: Game Design

- Mechanics/Technology
  - The game loop for this game is a simple loop that runs constantly until the player closes the game window
  - When the player traverses the platforms in the game and either gets to the next level or dies and has to respawn, a new level is loaded and the loop never stops. When new levels are loaded we simply switch between the overworld and the level as a type of menu where you can choose which level to play
  - The gimmick I used for this game was the amount of assets I used. I was able to get a full range of assets included the player character, three types of enemies, terrain tiles, SFX, and music
  - Not sure if my game differs from others of its genre however, I was able to implement a unique death animation for each enemy in the game
  - One of the strengths of my engine I believe is the organization of the files. There is a lot of code in this project, but all entities have a separate class for their implementations allowing for easy code changes and bug fixes
- Story
  - There is no story here, but the theme is that you are a sort of mysterious cloaked person/creature that is traversing the platforms in an outdoor setting in order to reach the end
  - The protagonist of the game is the main character whose only goal is to survive until they reach the end while the antagonists are the enemies that we run into along the way
- Player Experience
  - I wanted the players to have fun playing the game but also feel challenged and sometimes upset. For example, when they fall or die they restart the level from the beginning
  - There are currently no rewards in the game but the player can collect coins on their way
  - Just about everything in the game has sounds effects: jumping, killing, being hit, as well as some fun background music

Part 3: Game Design Changes

- In classic optimistic fashion, when I originally design the outline for the game I have planned for there to be things like enemies that attack you with projectiles, text voice lines for characters, a boss fight, and much more. However, with no only my lack of knowledge what a severe lack of time these things had to be cut one by one down to the current state of the game. While the general idea is still the same, it is a much more simple version of the original idea where enemies and the player have no attacks and simple movements and there is no boss fights or text for voice lines.

Part 4: Game Development & Documentation

- Outline
    - The “modules” for the game are discussed in the README on GitHub for the project here: https://github.com/Michael-Elrod-dev/Pygame-Engine
    - The Level class is the core of the project. It handles object initialization, terrain setup, asset paths, camera view, and controls the state of the game and player. This file also tracks all collisions between the player sprite, the terrain, enemies, and coins. The update function within Level.py which runs every frame and updates all sprites based on user input and camera movement.
    - The Player class handles the player's status by getting user input and updating the status of the player character based on its velocity. This file also calls the classes that import game assets.
    - The Enemy class handles enemy movement and animation changes
    - The Tile class handles the movement of tiles with the screen as well as animating tiles spaces like the coins
    - The UI class handles the interface and the Decorations class handles background images and things like grass and clouds
    - The Overworld class generates and overworld where the player can choose a level to play and see their progression
    - I also have a few more classes that act as support for the engine. For example, a Settings class that stores information about the game like the level map design (CSV) and the screen size.
    - A visual diagram of the MODEL-VIEW-CONTROLLER setup can be viewed on GitHub as well https://github.com/Michael-Elrod-dev/Pygame-Engine
- Bugs
    - There are currently no unintended bugs that I am aware of at this time however there are some things that are not technically complete, and I will list those here:
    - I had intended to make three separate levels, each level containing different enemies and harder terrain courses. However, because of the difficulty of my project I decided to stick to one level for now which included ALL 3 enemy types on it. This means that when the player reaches the finish line and continues to the next level it will just be the same exact level every time. This is why in the overworld each stage has an image of different enemies over it.
    - For some reason when I run this game on a Linux machine the music files don’t work right and animations do not play properly. For example, the animations that should only play one time like deaths, play many times over. This does not occur on Windows and I don’t know why. I still wonder if this would occur on a Mac device too as my code is not flawed and should work either way. This issue is the sole reason my game had to be simplified to this point because I spent weeks on this bug only to find out by coincidence that it was only occurring on Linux and therefore I confirmed that life is a meaningless void of pain and suffering.
    - I used Github for this project but I worked solo so there was no collaboration

Part 5: Division of Labor

- Michael Elrod – responsible for literally everything
- Rough time estimate of 40-50 hours
- Milestone 1: March 29th - All Completed
  - Michael – Get the map tiles generated in game by using a csv file generated from Tiled
  - Michael – implement Enemy and Coin sprites and their automated movements in game
  - Michael – Restrict enemy movements by using constraints in csv file
- Milestone 2: April 12th
  - Michael – Get my own sprites for enemies instead of stolen ones
  - Michael – Configure a UI system of some sort (Coins, Health, etc.)
  - Michael – Implement death animations for enemies
  - Michael – Implement player damage system
  - Michael – Implement a win/death state
- Milestone 3: April 26th
  - Michael – Implement SFX/Music
  - Michael - Complete game documentation
  - Michael - Check for bugs
- Game Presentation: May 4th
  - Michael - Learn how to make an executable
  - Michael - The game will be in a playable state for presentation

Milestone 1 Notes:

During this milestone I added the following implementations:

Enemies: I've imported some stolen sprites as a place holder for enemies. I also implemented their movement system to have them walking back and forth. This movement is restricted by checking for collision with invisible tiles that are located in my CSV file

Tiles: Using the Tiled software I created a 2D map and generated a csv file for that map. To decorate it I used some stolen sprites. This implementation takes a PNG of the tile set and splits the image into 64x64 sized pieces and places them where their respective numbers are according to the CSV file
Coins: I added some rotating coin sprites for fun. Currently they just spin and cant be interacted with but soon the player will be able to collect them!

Updated Timeline:
Nothing got moved up or down in the timeline but I did add some new features because I had some extra time. For example, I added some more in-depth UI stuff and added a possible earlier timeline for SFX. Also I added a death/win state condition to the third milestone.

Challenges:
Was massively challenged by trying to figure out how to work the Tiled software and get it to be configured properly for importing it to my game. To simplify this process instead of making several CSV file layers for different objects I have everything compressed into one singular CSV file. Definitely isn't ideal because this means no items can be stacked on game initial game state. If I wanted to add decorations like grass or trees or something, then my current implementation wouldn't allow that. The

CSV file would need to be separated into layers. If I don’t change this at a later time then the timeline wont be affected but the final “look” of the game will suffer as it will have a more plain appearance.

Milestone 2 Notes:

During this milestone I added the following features to my game:

- A UI system with a health bar and a coin count
- An overworld where the player can choose which level to play and see their progression
- A damage system for the player
- A death state that triggers respawning
- A win state that progresses you to the next level
- Reorganized the file structure to better group implementations with their respective classes for animations and importing
- Reformatted the way I take in a CSV file to allow for multiple layers of object on top of each other
- Created new assets for the games overworld, end goal and enemies using Aesprite

Updated Timeline: Nothing necessarily got moved in the timeline, but I was able to add more details to it for the current milestone

Challenges: The biggest challenge for me this time around was the death animations for the enemies. So much so that the bug I encountered is still present and probably will be by the time the game is finished. I have an implementation that is supposed to run a set of PNG files on screen looping just one time when an enemy dies to simulate their death. However, this animation plays several times before stopping. To be honest I’m starting to believe that only God knows why and a mere mortal like myself is not worthy of this knowledge and I am meant to be ignorant forever.

Milestone 3 Notes:

During this milestone I added the following features to my game:

- SFX and music
- Finally fixed the bug that was plaguing all my animations
- Learned how to create an executable file

Updated Timeline: I had to remove the “create new levels” portion of my timeline for this milestone because of the amount of time it took me to fix the animations bug

Challenges: The biggest challenge for me this time around was again, fixing the animations bug. This alone took me roughly a whole week and was fixed by mere coincidence.