Game Development with Python Using Pygame

Boyao Song[1]

[1]bobbypotato233@gmail.com

[1]Liaoning Institute of Higher Education, Shenyang, China

Abstract: When doing research, we found out that there are few works about writing a 2d game in a short time period. So, we came up with the idea of completing this work. In this work, the team go through the process of making a simple parkour game from scratch. When choosing the type of our game, the optional of making a parkour game is chosen as it is thought to be the most challenging topic. Because it is difficult to adjust the various constant like gravity constant in our game to make the game comfortable to play. This paper talks about different steps of game making including gathering assets, the usage of MVC model, making a physics system and the traps to prevent player from the finish line. I think the largest gain from this project is that to prove a game can be done in only few days by a limit amount of people.

Keywords: Computer science; game making; python programming language

1 Introduction

Game development with Python using pygame

In the year of 2020, Python has become the fourth most popular programming language [1]. There're also a lot of video games developed using python e.g., Doki Doki Literature Club and part of Battlefield 2 [2]. So, we came up with the idea of recreating a game using python within a short time period. After researching on GitHub and python wiki, the team found out that it would be good idea to make a mario-like parkour game because no one have done it before. So, the target of this work is to make a 2d parkour game like mario[8].

2 Method

2.1Main File

In the main game file, there is only initializations of different modules and the main game loop. The team decided to use MVC model [3] in order to increase working efficiency and decrease the amount of work in the debug processes. So, the modules are written separately in different files and folders and imported at the head of the main file. A start screen is also added to make the work more formal.

ICIDC 2022, June 17-19, Qingdao, People's Republic of China

DOI 10.4108/eai.17-6-2022.2322877

Figure 1 Main Page

2.2Asset Gathering

There're three parts of assets in the game. The background pictures, character pictures and the background music (Figure 2).

The background picture is made by splicing an asset picture found on an open-source website (Shown in Figure 3). An example of background is shown in Figure 4.

Figure 2 main character and enemy

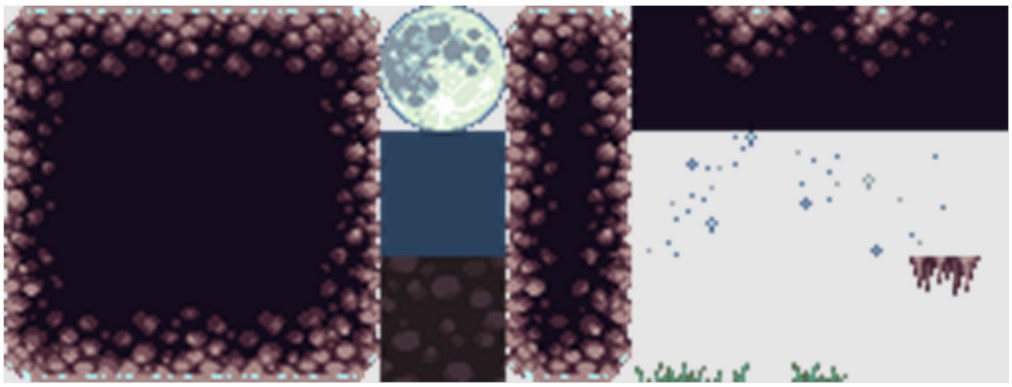

Figure 3 Asset picture [6]

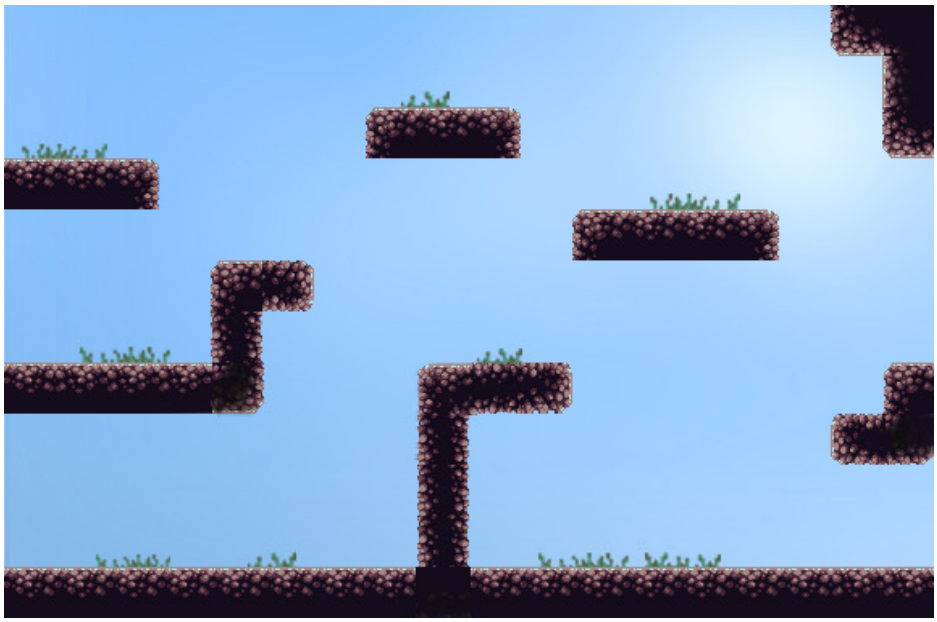

Figure 4 One of background pictures [6]

About the character pictures, the picture of a boss from a 2d shooting game called hollow knight is used as player picture (shown in figure 3) and also the monster picture as monsters from the same game. We also made a start page which is in minecraft style (shown in figure 1)

2.3Physics System

Collision detection is achieved by creating a class called physic and write different functions in it so that data can be stored in the class to make the file tidier. The whole physics system is separated in to two parts: falling and collision detection.

2.3.1Falling

To process the falling speed, a private method is created to calculate in-air-time so the speed of falling can increase by seconds like it works in reality. The render time of last frame is input into the function to calculate the in-air-time. The distance of falling in each frame is calculating by the formula of acceleration (shown in formula 1).

$$\frac{1}{2}gt^2 \tag{1}$$

Formula 1

The vector of falling is then added to the vector of character's location (shown in code 1) [7].

```
def main():
    start_time = time.time()
    # pass
    y -= physics_class.fall(end_time-start_time)
    # pass
    end_time = time.time()
```

Code 1

Where g is the gravity constant which is a variable defined in the head of file and t is the in-air-time calculated by the private class.

2.3.2Collision Detection

The whole map is seperated into an 18*12 blockmap (shown in Figure 5) to detect if there is block around the player. If there's block around the player, they are sorted into different direction and solved by the correct module.

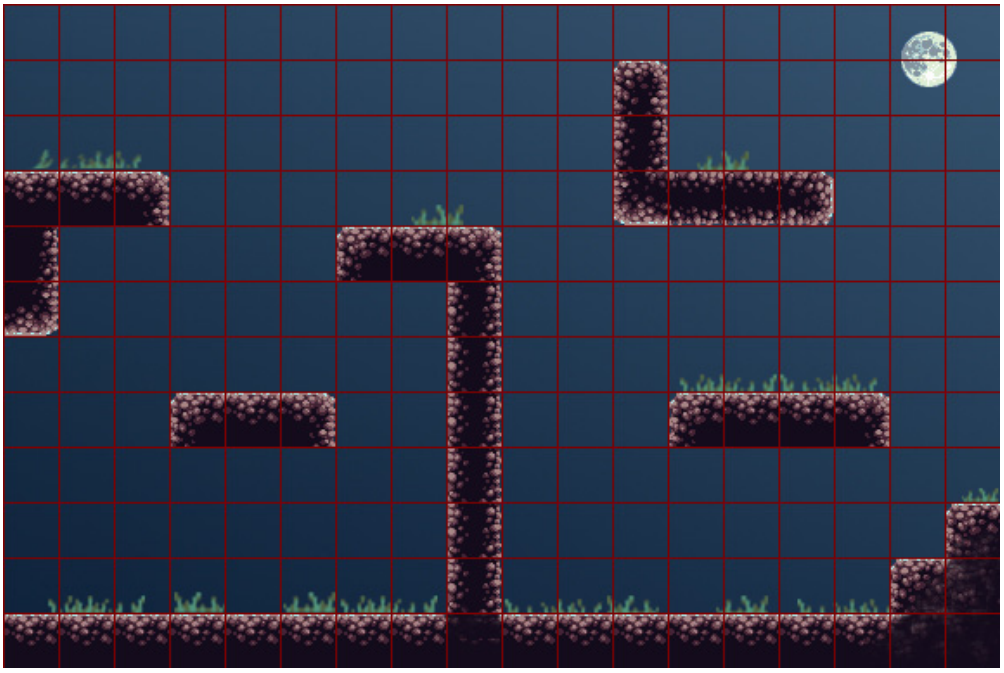

Figure 5 debug_map1

For the head and foot detection, the y-axis doesn't change if it's detected that there is block above head or under foot. It the same for side detection and the only difference is the axises.

2.4Rooms and Map

Instead of making a map list containing different rooms with different indexes. A parameter called room_id in generated under each instanced player class. The location of dirt blocks is stored in a dictionary called terrain_location under each map id so that they can be accessed at fastest speed (as shown in Code 2). [5]

```
terrain_location = {
    (0,0): True,
    # pass
}
```

2.5Monsters

At the early stage of this project, the team wanted to create mobs that can shoot bullets and chase the player. But as the deadline approaching, it is only possible to make it running in a fixed path and kill the player when player step on them or touch them (as shown in figure 6). As the same format as the location of dirt blocks, the monster location is also written in the room file in dictionary under each map id to achieve a lower time complexity.

Figure 6 the first game scene

2.6Rendering

For the rendering process, a dictionary that with the format of { (x, y): object, }, where x and y are the coordinate and object are the character, is passed into render. For example:

include. renderer. render(surface=surface, n={(c.x, c.y):c.status_avatar}) #[5]

Where surface is the background that player is currently at and n is the dictionary above.

3 Results

The completed project is shared and open-sourced on https://github.com/akaigua/some-2d-shooting-game.

4 Conclusion

In this work, the main framework on how an individual game developer code a game is concludes. I think the most impressive part of this work is a parkour game can be done in such a short time. I hope in the future, this article can inspire more individual game developers to tell stories about themself.

References

[1] *Stack overflow developer SURVEY 2020*. Stack Overflow. (n.d.). https://insights.stackoverflow.com/survey/2020.

[2] *Page*. OrganizationsUsingPython - Python Wiki. (n.d.). Retrieved September 13, 2021, from

https://wiki.python.org/moin/OrganizationsUsingPython.
[3] Wikimedia Foundation. (2021, October 26). Model–view–controller. Wikipedia. Retrieved November 16, 2021, from https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller.
[4] Python software Foundation Wiki Server. Python Software Foundation Wiki Server. (2020, August 18). https://wiki.python.org/.
[5] Pygame front page¶. Pygame Front Page - pygame v2.0.1.dev1 documentation. (n.d.). https://www.pygame.org/docs/.
[6] ZomBCool on 15 July 2021 - 4:25pm, MedicineStorm on 7 March 2021 - 1:39pm, MedicineStorm on 18 November 2020 - 6:50am, Botanic on 13 May 2020 - 4:31pm, & Xom Adept on 30 September 2019 - 11:21am. (n.d.). OpenGameArt.org. Retrieved September 13, 2021, from https://opengameart.org/.
[7] DvoleDvole 1, FxIIIFxIII 3, IainIain 6, & Loren PechtelLoren Pechtel 99244 silver badges88 bronze badges. (2011, September 10). How to calculate falling and accelerating velocity? Game Development Stack Exchange. Retrieved November 16, 2021, from https://gamedev.stackexchange.com/questions/14850/how-to-calculate-falling-and-accelerating-velocity#:~:text=compute%20the%20acceleration%20vector%3A%20A,V%20%3D%20V%20%2B%20A%C2%B7dt.
[8] Nintendo. (n.d.). The official home of Super Mario™ – home. The official home of Super Mario™ – Home. Retrieved November 16, 2021, from https://mario.nintendo.com/.