INTERNATIONAL JOURNAL OF SCIENTIFIC RESEARCH IN ENGINEERING AND MANAGEMENT (IJSREM)
VOLUME: 06 ISSUE: 12 | DECEMBER - 2022 IMPACT FACTOR: 7.185 ISSN: 2582-3930

Flappy Bird Game

Kavita Ugale, Parth Shethji, Kartik Parsodkar, Hitesh Pariani, Aditya Partole, Ritiksha Pardhi

Department of Engineering, Sciences and Humanities (DESH)
Vishwakarma Institute of Technology, Pune, 411037, Maharashtra, India

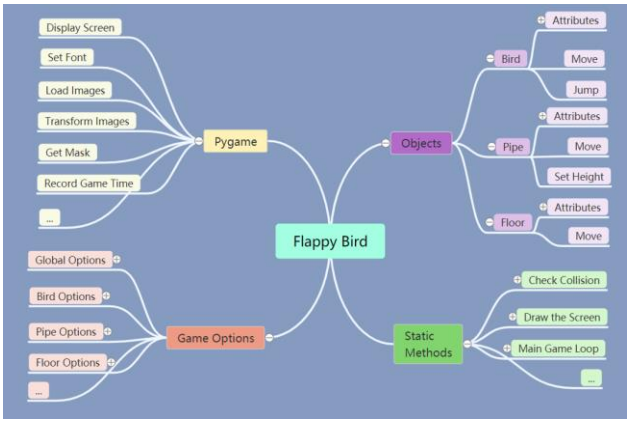
Abstract — We are all accustomed to playing this game. The player's primary goal in this game is to protect the bird from obstacles while racking up as many points as possible. Here, we'll use Python to create our own Flappy Bird game. To make this Flappy Bird game, we used the Python module Pygame. An open-source package called Pygame is made specifically for creating video games. It aids in the development of multimedia and fully functional games in Python.
Keywords:- Python, pygame, controls, collision, score

I. INTRODUCTION

Owing to the increasing use of smartphones over the past ten years, mobile games have significantly increased in popularity. Due to their addictive and amusing qualities, numerous games have gained prominence over time. Around the time of its introduction, thousands of people played Flappy Bird, which was one of the most well-liked games ever. The game was first released in 2013, but its inventor deleted it later that year out of shame about the growing addiction. Numerous versions of the game have been released online as a result of its popularity, and they continue to be well-liked users of the console version. Dong Nguyen, a Vietnamese videogame programmer, is the creator of the popular mobile game Flappy Bird. Similar to the well-known video game Super Mario Brothers, the side-scrolling game Flappy Bird has the user controlling a bird in a 2D setting. The player can only influence the bird's vertical movement, unlike Mario. The bird's main objective is to move as far without colliding with any green pipes as it can. The game's objective and controls are straightforward, but as the bird moves quicker the further it goes, the complexity and franticness of the game's gameplay increase. Python is a computer programming language used for various software programs, and when combined with resources like Python Arcade, it may be used to develop online video games. One of the games that can be made with Python is Flappy Bird, and if it is designed properly, it can be a near-perfect replica of the original. We developed a game design strategy for our Flappy Bird project throughout the implementation stage, with checkpoints every two to three weeks. In the early phases, a sound plan had to be established, the code had to be divided up, and a framework had to be built by using simple programming. Later stages required building on the foundation by introducing more obfuscated code, adding auxiliary components like sprites, colors, and lighting, as well as adding the game's finishing touches.

Formerly, Flappy Bird could be played on a phone by tapping the screen to make the bird fly. If the bird touches any pipes or edges of the screen, the game is finished, and the player should restart. The space bar or the up key can be used to control the bird in the computer version of the game. The programming language we'll use is Python. Additionally employed will be a cross-platform set of Python modules specifically created for game development. There are libraries for graphics and sound that were made expressly for Python use. Pygame is suitable for creating client-side plans over that could later be compiled into a standalone executable.

II. METHODOLOGY/EXPERIMENTAL



The necessary modules are being introduced. Random will be utilized to create random numbers for the purposes of our game. In the sys module, invoking sys.exit will lower the deficit. In lines 3 and 4, we are importing Pygame and the basic Pygame imports, respectively. After that, we declare a variety of game-related global variables. First, the fps, screen width, and screen height values are set. The screen is then created by passing the screen width and screen height as parameters to the pygame.display.set mode() function. Then, we construct a ground-y variable

that will supply the y-coordinate for our base image as well as two dictionaries, game pictures and game sounds, which will house the various images and sounds utilized for the game. The player (a bird), backdrop, pipe, and title images are then stored in these variables by giving their paths. The main method will be where our game launches, and it will need that all pygame modules be configured using pygame.init (). In order to keep track of time, we also construct the fps clock variable and use the pygame.tick.Clock() function. The photos will then be allocated to the "numbers" key in the game images dictionary after being initially stored in a tuple and given a title. Using pygame.image.load(), convert alpha(), and the paths to the pictures as parameters, we may transform an image's pixel format, including per-pixel alphas. The images of the message, base, pipe, backdrop, player, and title are added to the dictionary using various keys in a similar fashion. Additionally, we added a picture of an inverted pipe for the pipe by turning the image by 180 degrees using the pygame.transform.rotate() function. In order to add the sounds to the game's noises dictionary after that, we use a variety of keys. In a manner similar to how we handled photos, the pygame.mixer.Sound() function, which saves the sounds, is given a list of paths to various sounds as an argument. both mainGame() and welcomeScreen() which are defined in the following sections, are then called as part of a loop.

Now that we know the welcome screen will appear when the game starts, we can define the welcomeScreen() function. For the player, message, and title images, we start by figuring up their x- and y-coordinate values. As we selected the arguments using a hit-and-miss process, you can alter the values to best fit your needs. This section contains the base's x-coordinate. We then start a while loop, which always returns True, and we start a loop that won't stop unless the control tells it to. In this case, we employ the pygame.event.get() function to examine each event that takes place during the game. After that, we use the escape key to confirm that the game window will exit whenever a cessation event occurs. The next situation, whether we pressed the up key or the space bar, will be investigated. If so, we'll adjourn the gathering before returning to start the game. In addition, since no keys or buttons are pushed, the welcome screen appears. The backdrop, message, player, base, and title pictures will be positioned using the screen.blit() function. We will use pygame.display.update() to update our window and update our clock variable with the fps number as an argument in order to view only 32 frames per second.

As part of the implementation of the mainGame() function, which also includes once again providing the player picture and base coordinates, the variable score is now initialized to 0. Then, we build 2 pipes for bletting on the screen by using the getRandomPipe() function, which will be defined later. The x and y coordinates for the lower pipes and the inverted upper pipes are then listed. Once more, we chose values using the hit and trial method. Then, we created variables to represent the bird's various directions of motion. We also provide an acceleration variable. PlayerFlapVel is the player's flapping velocity, and Player Flapped is set to false (which is true only if the bird flaps. Then, we do another event search. First, we look for game-ending events, and if any are found, we end the game..

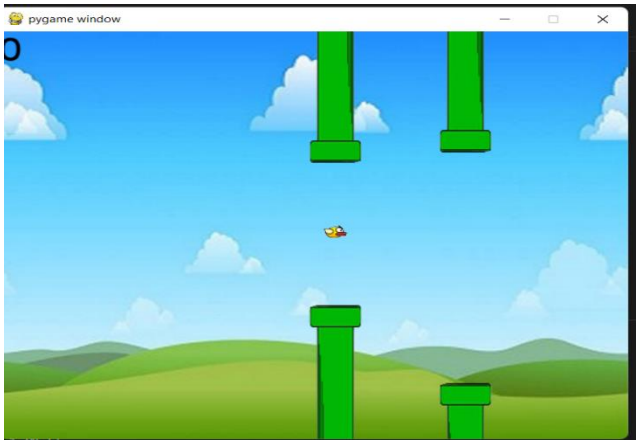
Then, it is identified whether the up or spacebar key was used. If so, we check to see if the player is below the top of the screen, at which point we update the game and activate the wing's sound using the.play command ().

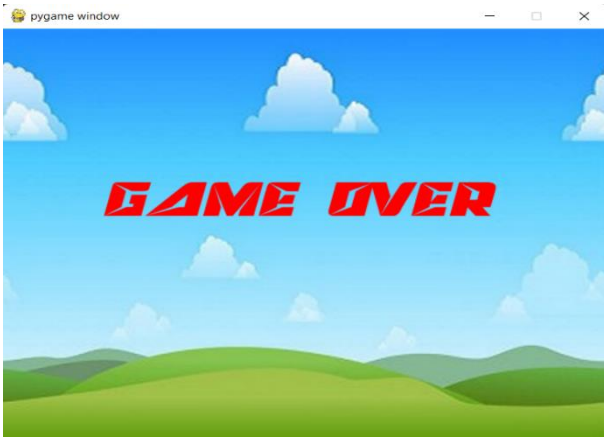
The next step is to check if we have crashed using the isCollide() method, which we shall define shortly. If the criterion is met, the function will end. The results will then be examined and revised. In the event that we cross a pipe, the score is increased, printed in the console using the player's mid position and the pipe positions, and the point sound is also played. If the player's velocity in the y-direction has not yet reached its maximum, we will then provide the acceleration. Later updates include updating the position of the bird and the playerFlpped value. We move the pipes to the left and add a new pipe when the first pipe is ready to cross the far left corner of the screen. We'll also see if the pipe is outside the boundaries of the screen; if it is, we'll remove it, add our own pipes, show the results, and then update the display screen. If there are more than one digits in the score, we first access all of them before inserting the required photos. We revise our clock one more time.

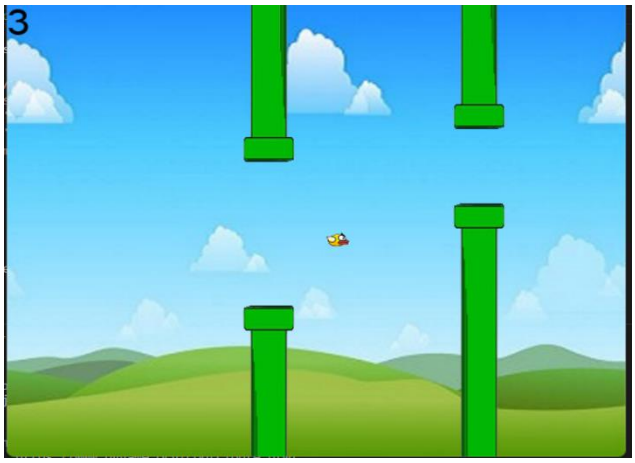
In the isCollide() function, we evaluate the position of the bird to the position of the pipe to see if we have reached the top of the base inline before looking for collision with upper pipes. The procedure is then repeated for lower pipes. If any of the collision conditions are achieved, we play the hit sound and return True. We utilize the offset variable to store one-third of the screen width and the pipeHeight variable to store the pipe's height in the getRandomPipe() function. We deploy random functions to generate values for the x and y coordinates with identical distances between the pipes but varying sizes for the upper and lower pipes. We save the coordinates in a list called pipe before returning them.

III. RESULTS AND DISCUSSIONS

Designing a flappy bird game is simple and only requires the sys, random, and pygame modules. The game still has a few small flaws, such as instances where the pipes overlap or come very close to one another. In order to save scores permanently, we can also add a feature that allows to see previous High Scores displayed on a screen. We can include a start menu and different difficulty levels to make the game more user-friendly.







IV. FUTURE SCOPE

[1]. Add start screen
[2]. Add high score of entire game
[3]. Make the graphics a little more attractive and smoothen the conto**urs**

V. CONCLUSION

Python is one of the programming languages that is most in demand today and is used for a wide range of tasks, including data science, game development, and the creation of software and other things. These days, game programming is quite lucrative and may be applied to advertising and education. Game development involves many different subjects, including arithmetic, logic, physics, AI, and much more, and it can be a tone of fun. Python's pygame module is one of the best for game programming and is used for it. The goal of the game is to avoid pipes and earn the greatest score possible.

VI . REFERENCES

[1]. The dinosaur game on google when we are offline

[2]. Took reference from the original flappy bird game

[3] Also for coding and learning pygame and python we took reference from GEEKS FOR GEEKS, YOUTUBE

[4]https://www.geeksforgeeks.org/python-programming-language/

[5]https://www.geeksforgeeks.org/introduction-to-pygame