Final Project Proposal

Flappy Bird (Team 9)

Released in 2023.06.16

- Abstract:

This game serves as the Final Project for NTHU Programming I. Our theme is Flappy Bird. The main gameplay involves using the spacebar as the means of moving the character. When pressed, the character will fly upwards, and when released, it will fall freely (the gravity value for this part can be adjusted within the character structure). And, this project utilizes the concept of I/O for file read and write operations, allowing for recording and displaying the highest scores as a small additional feature and practice. Additionally, we have also implemented game medals as rewards when certain scores are achieved. Gold, silver, and bronze medals are awarded accordingly.

- Method of Implementation:

To achieve a fixed horizontal displacement for the character and have it move only within the same vertical plane while the pipes (obstacles) continuously approach from the left, you can follow these steps to simplify the implementation and avoid potential horizontal overflow of the character.

- Function Introduction:

Part 1

1. must_init() [code 22-27]: Ensure smooth execution of each code, and when an error occurs, immediately terminate the function and return an error message to prevent users from experiencing faulty game presentation during incomplete code execution.
2. Score() [code 29-62]: Use the concept of I/O to read and write the highest score, as well as determine if it is the highest score. If the score is updated, execute the function to update the score.
3. Display() [code 64-105]: This part of the function includes screen initialization, destruction and resource release, drawing on the buffer, and presenting the final result on the actual screen for improved execution efficiency. If you want to change the window size, you can modify the value of "Display scale" here, but it is not recommended to modify Buffer_W and Buffer_H, as it may cause layout issues.
4. Keyboard() [code 109-115]: Simply initialize the values of the button array in advance, to be used for subsequent event execution.
5. Sprite() [code 117-235]: Initialize the resources (image resources) present on all pages, perform construction, release resources, and so on.

Part2

1. Audio () [code245-286]: This part of the function is for loading and playing audio, after playing, you should clean it to release the memory spaces. In this step, the background sound plays without stop and you can adjust the playback speed and the volume as you like.
2. Drawing () [code 289-374]: Firstly, check the player's alive or dead and then drawing based on its situations. If the game is over, the window creates the scoreboard which includes the player's record. If you want to continue playing, the OK button will lead you till the game, otherwise, clicking Menu button will move you to the beginning of the game.
3. Player () [code361-378]: This part includes all things related to the character which include the coordinate, the situation and the player's drawing. You could modify the components of character as you want but it's not recommended since we fully optimized the best for you.

Part3

1. Obstacles() [code 382-448] : This part demonstrates the obstacle in this game is pipe. When the player touches the pipe, the player would die and the game ended. Also, when the player jumps over the range of the screen or drop to the floor, the player also died.
2. Main():
    - [code 452-489]
        - Create timer and event queue
        - Induce keyboard, display, timer and mouse function to event queue to run in the game loop.
        - Initialize spirit, font, keyboard, player, pipe and audio function.
    - [code 492-656] This part demonstrates the process of playing the game. When pressing the key-space button, the game starts. When the game starts, updating the gravity and pipe. If the player dies, pressing "ESC" and renew the parameter of the game. This part also demonstrates the condition of pressing and releasing the button, and the best score that the player gets.

➢ Target Platform

Since this work is programmed in the C language, it can be smoothly executed on various system platforms. However, it requires preparation of the runtime environment beforehand. To do so, simply open the folder and execute the Makefile.

➢ Development Tools to be used

1. Main Language: C programming
2. Graphics Library: Allegro5
3. Programming Editor: VScode for MacOS & CodeBlock for Windows

- Group coordination

Part 1 (林英豪): Team Leader ,Main architecture, create Makefile, asset creation, and program initialization development.

Part 2 (Kian): Audiovisual design, user interaction, character configuration.

Part 3 (蔡宥勳): Obstacles, key events, character death events.

- ✧ Each team member is allocated an equal proportion of 33.3%, with no possibility of slacking off or plagiarizing.

- How to play:

Download ZIP from the link mentioned below -> drop it into VScode -> Open your Terminal in this Folder -> cd flappy/src -> make -> ./game.out -> Enjoy your Game.

- More detail:

Detailed information about code resource packages, production logs, task allocation sheets, learning reference materials, how to adjust the difficulty level with some parameter, and more can be viewed from the following link.

(You can find something new in ReadME.md)

https://github.com/SCLemon/NTHU_I2P_Project