CAPSTONE PROJECT REPORT

for

Flappy Bird Remake

ThanhTH10

- HCM City, October 07 2024

Contents

1. Introduction

Project Overview:
The project is a 2D remake of Flappy Bird, developed using **Qt 6** and **QML**. The objective is to guide a bird through obstacles (pipes) without colliding with them. The game records the player's score and saves it in a local SQLite database, with the aim being to achieve the highest possible score by successfully passing through as many pipes as possible.

Objective:
The main objective is to develop a fully functional 2D game using **QML** and **SQLite** for score management. The game should implement smooth animations, user input handling (keyboard and mouse), and persistent storage of high scores.

Scope:
The project covers basic game functionality such as:

1. Bird movement and gravity simulation
2. Pipe generation and collision detection
3. Score calculation and display
4. Persistent storage of high scores using SQLite
5. Displaying a leaderboard with top 5 scores.

2. Development Tools and Technologies

Programming Language:
The project uses **QML** for UI design and **JavaScript** for game logic.

Framework: **Qt 6 -** Used for developing the game, handling UI, animations, and signals.

Database: **SQLite**: A lightweight database used for storing high scores, including the score and date/time.

Other Tools:

- **Qt Creator IDE**: The integrated development environment used for coding, debugging, and building the project.
- **Git**: Version control system for managing the source code.

3. Game Design

Game Mechanics:

- **Bird Movement**: The bird moves vertically in response to user input (keyboard or mouse). Gravity is applied to the bird, causing it to fall continuously, and the player must press spacebar or click the mouse to make the bird jump.
- **Pipe Obstacles**: Pipes are generated at regular intervals and move horizontally across the screen. The player must navigate the bird between the gaps in the pipes.
- **Game Over Condition**: The game ends if the bird collides with a pipe or the ground. Upon collision, the player’s final score is saved in the SQLite database.

- **Score Calculation**: The score increments each time the bird passes a pair of pipes. The current score is displayed during gameplay, and the high score list is shown at the end of the game.

Graphics:

- **Background**: A static background with repeated tiles to simulate infinite scrolling.
- **Bird**: An image of the bird sprite that moves along the y-axis.
- **Pipes**: Rectangular images representing the obstacles that move along the x-axis.
- **Ground**: An image representing the ground, located at the bottom of the screen.

Animations:

- **Bird Jump**: When the player presses the spacebar or clicks, the bird jumps upward with an animation.
- **Pipe Movement**: Pipes move continuously from right to left, simulating the forward movement of the bird.
- **Background and Ground Movement**: The background and ground tiles scroll horizontally to create the illusion of movement.

Sound:

- Background music can be played using the SoundEffect component to enhance the player's experience.

4. Architecture and Implementation

Game Structure: The game is organized into several components:

- **Main Window**: The main Window element holds the game environment.
- **Game Environment**: Includes the bird, pipes, and background images. The game logic is handled within a Rectangle that encompasses the entire game area.
- **ScoreManager**: A custom QML component that handles saving and retrieving scores from the SQLite database.
- **Animations and Timers**: The game relies on QML's Timer and SequentialAnimation components for smooth transitions.

Score Management:

- **SQLite Integration**: Scores are stored in a local SQLite database. The database schema includes fields like Id (auto-increment), Score, and DateTime.
- **saveScore() Method**: Called when the game ends, this method saves the current score to the SQLite database.
- **fetchTopScores() Method**: Retrieves the top 5 scores, which are displayed in a list when the game is over.

Game Logic:

- **Bird Control**: The flappy() function is triggered by a spacebar press or mouse click. It animates the bird's upward movement and controls rotation.
- **Collision Detection**: The bird's position is constantly monitored using the onYChanged event to detect collisions with the ground or pipes.

- **Game Termination**: The terminateGame() function is invoked upon collision, stopping all animations and saving the score.

Animation and Timers:

- **Pipe Creation**: The pipeCreator Timer triggers the creation of pipes at regular intervals (2.5 seconds).
- **Gravity Simulation**: A Timer continuously moves the bird downward, simulating gravity.
- **SequentialAnimation**: Used for bird rotations during jumps, making the game visually engaging.

5. Database Integration

SQLite Database: The SQLite database stores high scores persistently. The database is initialized when the game starts.

Table Schema:

- Id (Primary Key, auto-increment)
- Score (Integer)
- DateTime (Date and Time when the score was recorded)

Score Saving: The saveScore() function inserts a new record into the SQLite database when the game ends.

Fetching Scores: These scores are displayed in a ListView upon game over.

6. Challenges and Solutions

Game Logic:
One challenge was ensuring smooth collision detection between the bird and pipes, especially when the bird moves quickly or at varying speeds. This was resolved by implementing continuous monitoring of the bird's y-position and the pipe's x-position.

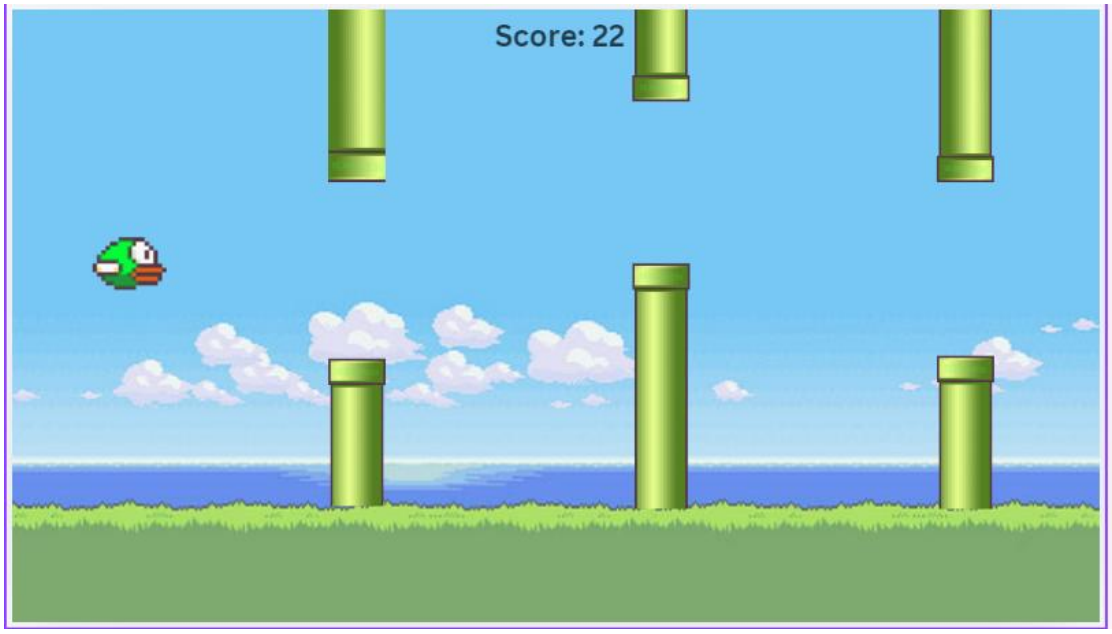
SQLite Integration:
Another challenge was ensuring that the SQLite database interacted seamlessly with the game in real-time, without causing lags during gameplay. This was achieved by optimizing the database queries and only saving scores when necessary.

Graphics and Animations:
Achieving smooth animations for the bird's jump and rotation while ensuring consistent performance was challenging. By fine-tuning the SequentialAnimation properties and easing functions, this was effectively resolved.

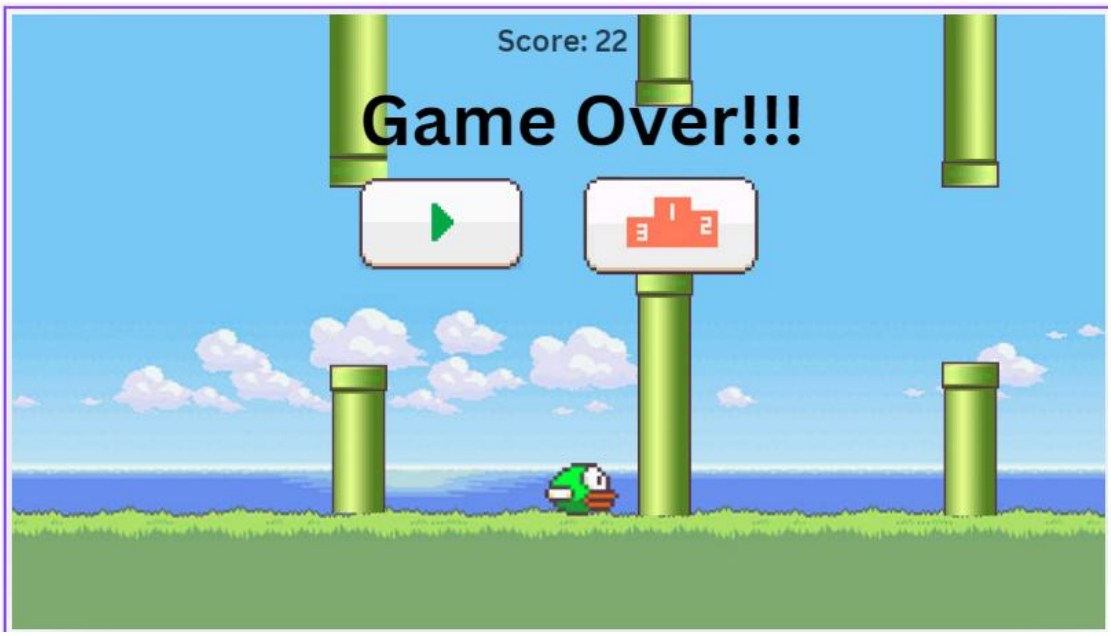
7. UI Design

- Game landing

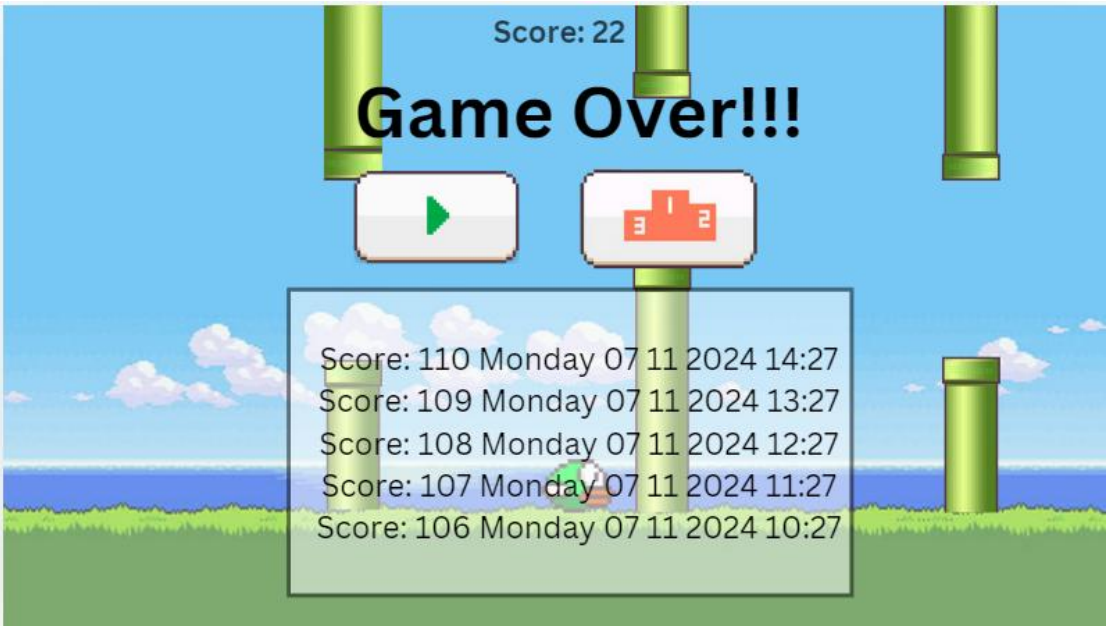
- Game play



- Game over



- Score:



8. Game Deploy

- gitlab: https://gitlab.com/mlananhh/qt_qml_finalprojec_flappybird
- github: https://github.com/moclananhh/QT_QML_FinalProjec_FlappyBird

9. Game Demo

- Demo game: https://drive.google.com/file/d/1AUfUFHFF_pjTY2VDWeROqiL5TL3eKKb-/view?usp=sharing