

Qt Creator: Scrambler

Mostakim Sifat
Computer Science
Auburn University at Montgomery
Montgomery, AL
msifat@aum.edu

Christopher Williams
Computer Science
Auburn University at Montgomery
Montgomery, AL
cwill208@aum.edu

Abstract— This project presents a trivia based version of Wordle created with Qt Creator in C++. The game gives players a trivia question and asks them to guess the correct five-letter word. Players type their guesses into a row of boxes, and the game uses colors to show if letters are correct, in the wrong place, or not in the word at all. A countdown timer adds challenge, and each correct answer increases the game level. The program also includes a leaderboard that saves the top five scores to a text file so players can see how they compare to others

I. INTRODUCTION

Word games are popular because they are enjoyable and help improve problem-solving skills. A well-known example is *Wordle*, which uses simple guessing rules to engage players. However, several word games do not change in difficulty as the player progresses. This can make the game feel less challenging and reduce the player's interest over time. To solve this problem, this project presents a word puzzle game made using Qt Creator.

In this game, players respond to a continuous sequence of word-based questions while managing a countdown timer. Instead of progressing through predefined levels, gameplay extends as long as the player continues to answer correctly. Each correct answer adds additional time, encouraging accuracy and quick decision-making. The game session ends only when the timer reaches zero or when the player exhausts all available chances.

An important part of this project is the use of queues in the game's design. Queues are used to manage and organize the sequence of questions in a structured design. This ensures smooth progressions between levels and a system for handling game data. The primary goals of this project are: (1) to develop a clear and user-friendly interface, (2) to implement a gameplay model that scales with player progression, and (3) to integrate queues to organize question sequencing and overall game flow effectively.

II. LITERATURE REVIEW

A. Artle – National Gallery of Art

Artle is a game created by the National Gallery of Art where players guess the artist based on a picture of a painting. Artle uses a clue-and-answer style that makes players rely on what they already know rather than only guessing letters. Our game uses a similar idea but focuses on general knowledge instead of art. Instead of showing an image, the game gives a short question, and the player has to remember a five-letter answer.

B. Wordament – Microsoft

Wordament, a game created by Microsoft, is a fast-paced word-search game where players race to find as many words as possible in a shared letter grid before the timer runs out. It focuses on speed, quick pattern recognition, and competing with other players in real time. Our game is similar in the sense that it also uses a timer to keep the player engaged, but the challenge is different. Instead of having the players search for the words, the players answer a question by recalling a specific five-letter word.

III. METHODOLOGY

Our game, Scrambler, was developed in qt Creator using C++, with the intention of creating a trivia-based version of Wordle. Instead of guessing a hidden word, the player answers short general knowledge questions, which with a five-letter solution. The Qt Widgets framework was used to build the interface, including the timer, the guessing grid, input handling, and the different screens of the application.

The project uses a text-file system to store player scores. A file named *leaderboard.txt* is maintained in the project directory. The game reads this file at startup using QFile and QTestStream to load the top five names and scores. When the player finishes and chooses to save their result, the file is rewritten with the updated information. Since this file is included in the Github repository, the leaderboard is automatically saved online.

The data structures used in the game program were vectors, pointers, queues, and a priority queue. Vectors held the questions, the answers, the list of used questions, the player names, and the scores. A queue is used to control the flow of questions during the game. When the program starts, question indexes can be pushed into the queue, and each time the player gets a new question, the game pops the next index from the front of the queue. This structure makes sure the questions are handled in order, and once a question is used, it is not selected again. A priority queue is used for ranking the top scores. When player earns a score, that score is pushed into the priority queue. The structure then keeps the scores in the correct order, so the program can pull the highest scores to display or save. Pointers also play an important role in the game. Each letter box in the guessing grid is a pointer to a QLabel on the screen. These pointers let the program change the text or the background color of any box when the user types or when the answer is checked. By storing these QLabel pointers in a two-dimensional vector, the game can update each row and column easily, similar to working with a small grid or table.

Gameplay is controlled by timers and keyboard input. A QTimer reduces the time bar once per second, and correct answers add more time. A question is taken from the queue, and the matching answer is pulled from the answer vector. As the player types, the current row of QLabel pointers fills one letter at a time. When the guess is checked, each letter is compared to the answer, and

the color of the box changes to show if it is correct, in the wrong position, or not in the word. The game ends when the time reaches zero or when the player uses all three attempts for a question. Any high score is then added to the priority queue, the leaderboard is updated, and the file is saved and backed up through GitHub.

IV. CONCLUSION

The result of this project was a fully working trivia-based game similar to Wordle that includes timed rounds, color-based feedback, a level system, and a saved leaderboard. Players receive a trivia question, type in a five-letter answer, and immediately see whether their answers are correct or not. The game also prevents repeated questions and keeps track of the high scores across sessions. Seeing how all of these pieces come together showed us that we were able to design and build a complete, playable program using what we learned in class.

REFERENCES

- [1] Wardle, Josh, *Wordle*. 2021. The New York Times, <https://www.nytimes.com/games/wordle/index.html>
- [2] National Gallery of Art, *Artle*, National Gallery of Art, <https://www.nga.gov/artle>
- [3] Wordament, Microsoft Casual Games, n.d, <https://www.microsoftcasualgames.com/wordament>