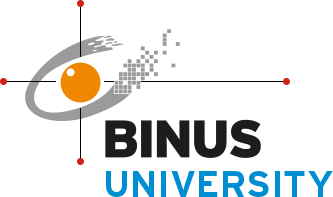
Word Games App

OOP Final Project



Object Oriented Programing

By: Hilkia Kennan Latjandu

NIM: 2602174485

Computer Science

1

Table of Contents

Table of contents………………………………………………………………………………1

1. Introduction…………………………………………………………………………………2

2. Inspiration……………………………………………………………………………..........2

3. Project Specifications……………………………………………………………………….2

4. Sources……………...………………………………..…………………………………….4

5. Class Diagrams……..………………………………………………………………………5

6. Evidence of Working Program…………………………………………………………….6

7. Lesson Learned……………………………………………………………………………8

8. Project Codes and Presentation Video……………………………………………………..8

2

Introductions

I would like to Introduce myself, I am Hilkia Kennan Latjandu, a college student of Binus International University 2026 for Computer Science my Student ID is 2602174485. This report was made with the purpose to explain and document my Final Project for the course Object Oriented Programing.

This Project is a program using OOP and uses Java Language. This Project is a compilation of word games. This Project is reliant in the txt file provided and uses Java Swing for its user interface. This means that the code uses txt files to get the words for the word games and to keep the scores of each game, and Java Swing is a GUI toolkit that is a little complicated to use, compared to JavaFX. The purpose of this Project is for a score for OOP. The purpose of the Code is to create a more simplified version of multiple word games and a app/code that is like a compilation of those word games.

Inspiration

I chose to do this because while I was trying to think of what a good code to make as my Final Project, I saw one of my friends playing a sort of word game. I was also inspired by my previous attempt to create a compilation of board games that unfortunately had issues, so I decided to do something similar, and I wanted to make sure that I can make it without issues. Thankfully in this project I managed to do it without any issues when finished. I used Java Swing because it is easier to set up in the early phases, but at a certain point the locations of buttons, etc. became more3 complicated, thankfully it functions.

Project Specifications

In side of my Github contains 8 Java files :

1. The file called game, is just basically for the interfaces.
2. The first file for UI is Main (start page), the file is basically a driver that runs the first UI that is seen when the app/game is run. The UI consists of just a simple Start Button that when pressed will run the next file which is options.
3. The second file is option (option page), the file is basically where the options or possible games are located. This file is just used as a middle ground to choose certain word games or to go back to the previous file which is main or the start page. Inside of this UI contains 5 buttons, 1 return button which when pressed will change the page into the start page. The rest of the buttons lead to the 4 (currently) word games which are hangman, scramble, unscramble, and wordle. Each button runs their respective games.
4. The next file is hangman (hangman game page), this file creates and runs all the UI for the hangman game. This file is the hangman game, which is a game where you guess the word that is randomly selected from the list of words in wordgames1.txt . the word is set as \* and every time a letter is inputted it will change \* into the letter if the original word contains the letter. In my program I made it so that it is possible to input letters but it is also possible to input words, and if the word is correct you

3 instantly win, but if it is not, it will check for the first letter of the inputted word. I have also implemented a checker that checks if the inputted word/letter has been inputted before, if yes then it will not consume an attempt. Each game contains 13 attempts, if you win or change all the \* to letters then you will get a point or point +1, and if you lose you do not lose or gain any score, win or lose it the correct answer will be shwn. The UI contains the word which is replaced with \*’s, a counter for your attempt and score, a textfield for input and a button that will input the text inside of textfield. In the bottom there are standard buttons back which will send you to options page, reset which will reset or make your score to be 0, and a restart button that only appears when the game has ended, even if you lose or win, that when pressed will re-run the code and reset the words, attempts, and the restart buttons visibility.

1. The next file is scramble, which is a kind of word game that I believe I made myself. Inside this game you are given a word from wordgames3.txt which contains around 10 letters, and you must make a word using the letters contained inside the word. The inputted words are checked if it is a real word by checking with wordgame2.txt, and if it is correct, your score will increase and if it is not, you will not lose any score. You start the game with only 2 attempts. The UI contains the word, the number of attempts, score, and a textfield for the input and a button that will input or take what is written in the textfield as the input. In the bottom there are standard buttons back which will send you to options page, reset which will reset or make your score to be 0, and a restart button that only appears when the game has ended, even if you lose or win, that when pressed will re-run the code and reset the words, attempts, and the restart buttons visibility.
2. The next file is unscramble, it is a basic game where it takes a word from textfield3.txt, and scrambles the order of the letters, and you win by inputting the word before it was randomized/scrambled. You start the game with 7 attempts, and for each failed attempt it will give you a random hint, it will say what letter is the n number in the word, where n is a random number less than length of the word, for example the 7th number is r. if you win score +1 and losing does not affect the score, but no matter win or lose the correct answer wil be shown. The UI contains the scrambled word, the number of attempts, score, and a textfield for the input and a button that will input or take what is written in the textfield as the input. In the bottom there are standard buttons back which will send you to options page, reset which will reset or make your score to be 0, and a restart button that only appears when the game has ended, even if you lose or win, that when pressed will re-run the code and reset the words, attempts, and the restart buttons visibility.
3. Next is wordle, which is a very standard wordle game. The game gets a 5 letter word from wordle.txt and you have to guess what that word is by inputting 5 letter words. The input will fill a grid of buttons that does nothing, but if a letter in the inputted word is inside of the word chosen then it will change the buttons color into yellow, if it also is in the correct index it will instead turn green. You are given 8 attempts, if you win then score +1 if you lose it will not affect the score, either way it will say what the word was. The UI contains the 8 x 5 grid of buttons, the number of attempts, score, and a textfield for the input and a button that will input or take what is written in the textfield as the input. In the bottom there are standard buttons back which will send you to options page, reset which will reset or make your score to be 0, and a

4

restart button that only appears when the game has ended, even if you lose or win, that when pressed will re-run the code and reset the words, attempts, and the restart buttons visibility.

1. The last is gamebackground, this file is basically the superclass that contains all of the imagery and buttons. It controls the background image and the button images. It controls which image, the layout, size, etc.. this file is important because if it does not exist there will be no buttons which means that there is basically nothing to do, and there would be no game.

Sources

background image:

https://www.pexels.com/photo/brown-wooden-parquet-flooring-129731/

button images (drawn by myself):

1. https://www.pixilart.com/art/try-sr28a3b453fd5b4

2. https://www.pixilart.com/art/start-sr2c8eb0a0d111f

3. https://www.pixilart.com/art/back-sr205c296e82e31

4. https://www.pixilart.com/art/reset-sr2463c2f3718a9

5. https://www.pixilart.com/art/restart-sr2c8c3f4729adb

6. https://www.pixilart.com/art/hangman-sr246c50ec4fe74

7. https://www.pixilart.com/art/scramble-sr2c2e158114c80

8. https://www.pixilart.com/art/unscramble-sr22cd71d79ee8c

9. <https://www.pixilart.com/art/wordle-sr293527fdc025b>

5

Class Diagram

uml diagram:

https://miro.com/app/board/uXjVMB3SYvE=/?share\_link\_id=951233841382

A screenshot of a computer

Description automatically generated with medium confidence

6

Evidence of Working Program

Start Page

A screen shot of a game

Description automatically generated with low confidence

Option Page

A screenshot of a game

Description automatically generated with low confidence

7

Hangman

A screenshot of a computer

Description automatically generated with medium confidence A screenshot of a computer game

Description automatically generated with medium confidence

Scramble

A screenshot of a game

Description automatically generated with medium confidence

Unscramble

A screenshot of a computer

Description automatically generated A screenshot of a computer

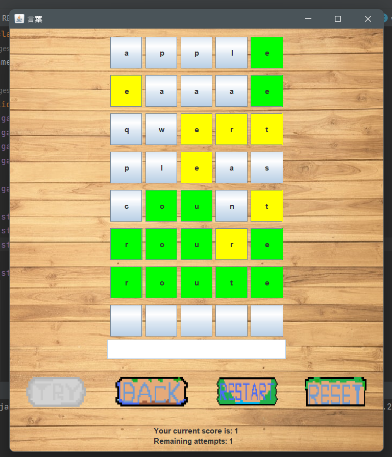
Description automatically generated A screenshot of a computer game

Description automatically generated with low confidence

Wordle

A screenshot of a computer game

Description automatically generated with medium confidence A screenshot of a computer

Description automatically generated 

8

Lesson Learned

I have learned how to apply OOP into actual code. I have learned how to read a text from a txt file and be able to use it in a code. I have learned how to use Java Swing.

From this I have learned that OOP is very useful for medium to large size projects, because it makes it easier and faster to code since I can use the previous codes without having issues that would arrive even if I used public. I have learned that compared to javaFX in the beginning Swing is faster but in the long run and if you would like to have a better end result/GUI then it is worth it to use JavaFX. Even so Swing has much better functionality even if it increases the complexity to understand it. I have learned that if I focus, be motivated and not aim for something incredibly difficult as I did in the first semester, then a compilation is fully possible. I have learned once again that creating games has a lot of small details that as a programmer, I have to really pay attention to. I have learned after testing that, atleast for testing I should not overcomplicate the end conditions of simple games, this is learned from the fact that I can barely win in these games myself, due to the high number of possibilities for the answers.

Project Codes and Presentation Video

Github Link:

https://github.com/F3Nwuff/OOPFinalProjectSem2/tree/main/OOPfinalproject

Presentation Link:

https://clipchamp.com/watch/k8ERtnNboOa