OBJECT DISORIENTED

Boggle Design Document

Version 2.0 – 2022.12.07

Originally Created 2022.11.06

Group members

Daniyal Bokhari
Sabrina Seiler
Sukhdit Saini
Xiaoyan Dong

SECTION 1: PROJECT IDENTIFICATION

- **Why are you doing this project (i.e. what is the motivation?)**

  To improve the boggle game (created in assignment 1), by implementing a user interface for the game. To visualize the game and make it better to play. Additionally, implementing features useful for the end user.

- **How will it enhance or add to functionality that already exists?**

  Firstly, creating a user interface to make the game more convenient to play compared to typing into the console. Secondly, incorporating features to make the game more accessible such as narrators.

SECTION 2: USER STORIES

Name	**ID**	**Owner**	**Description**	**Implementation Details**	**Priority** 1-lowest 3-highest	**Effort** 1-lowest 3-highest
Basic UI	1.1	Daniyal	As a user, I want to be able to see a window with a space for the game board and areas on the side with buttons	Use JavaFX to create a window with a board to display letters and buttons on the sides	3	3
Saving Statistic	1.4	Jacky Dong	As a user, I want the program to be able to save statistics of games that have been played	Create a file on the user's hard drive containing game statistics.	1	1
Loading Statistic	1.5	Jacky Dong	As a user, I want to load statistics of games that I have played.	Load a directory, if it is a loadable boggle saved file, load it for displays later.	1	2
Update Counting Score	1.6	Jacky Dong	As a developer, I want to upgrade the codes for counting scores by words found.	Use the visitor design pattern to implement this function.	1	1
Basic Setting Selection	1.7	Sabrina	As a user, I would like to see a window where I can select the board size and choose to randomize letters before the game starts.	Use JavaFX to create a separate view for the settings that pops up before the	3	2

				game starts. Use these settings to configure the board size and letter selection for the board		
Display Grid	1.8	Daniyal	As a user, I would like to see a grid displayed on the screen with buttons and letters on them so that I can see the boggle board I am playing on.	Using JavaFX canvas objects and button objects, create a grid with letters in each grid position.	3	3
Guess word button	2.2	Sukhdit	As a user, I would like to be able to press a button to submit the word I've guessed on the board and see if it is correct	Create a JavaFX button that reads the word the player has made and then checks if it's valid and awards them points	3	2
Hint button	2.3	Daniyal	As a user, I want to be able to press a button and get a hint for a word I haven't found, only the first and last letters and the length	Create a JavaFX button that calls on a function which displays hint for a random word that hasn't been found yet	1	1
High score	2.4	Sukhdit	As a user, I want to be able to see the highest score out of every round played by the player	Update a local variable *Max* at the end of every round and display it on the interface.	1	1
Get Custom Letters	2.6	Daniyal	As a user, I would like to input custom letters to be put on the game grid.	Use JavaFX to create a text input field where a user can enter the letters they want on the board, and initialize a grid with these letters.	2	2
Playable boggle game	2.7	Sabrina	As a developer, I would like the elements of the UI to be connected to the game code and allow the user to play a	Set the action of elements on the screen such as buttons to call methods in BoggleGame in order to play a game of boggle.	3	3

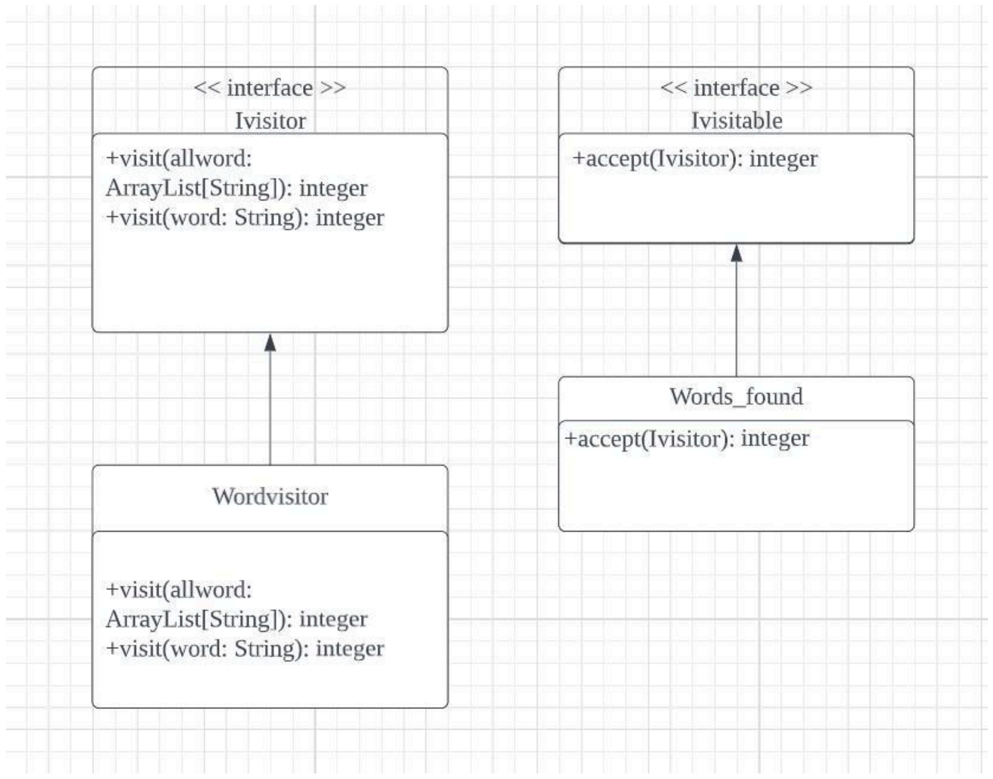
			round of boggle.			
Guess with buttons	2.8	Daniyal	As a user, I would like to guess a word by pressing buttons on the screen and seeing a line drawn between them to show the word I'm guessing	Use JavaFX to draw lines between buttons that have been pressed, and use a method to check if the buttons that are pressed are a valid word.	3	3
Word Counts	2.9	sukhdit	As a user, I may want the interface to display how many words are left unfound.	Create an object to store the amount of words unfound and update the count when a word is found	3	1
Score label	2.10	Sabrina	As a user, I would like to be able to see my current score in the game on the top right of the screen	Use JavaFX to make a text box on the top right of the screen that displays the current score using a method getPlayerScore() in BoggleStats class	3	2
Reading out Letters	3.1	Sabrina	As a user, they can click on a letter and the program will read it out loud.	Add a "read-out-loud" button, use voice-over for Mac and narrator for PC to read out the letter a user presses	3	2
Implement command pattern	3.2	Daniyal	As a developer, I would like to use the command pattern to issue commands such as guessing a word or restarting the game	Create an invoker class that contains commands and executes them. Make an interface for a command with an execute method, and implement concrete commands for guessing a word and restarting the	2	2

				game		
Singleton pattern	3.3	Sukhdit	As a developer, I would like to use the singleton pattern to display the available words that the player can still guess.	Create a WordCountSingleton class that contains the total word count and player word count number. Provide getter and setters for word count and player word count. Update it whenever the player guesses a correct word	2	1
Correct path	4.1	Daniyal	As a user, I would want the colour of the line between letters to be green if the word I'm guessing is a valid prefix to a word in a dictionary, and red if it is not	Use JavaFX to change the line colour depending on the letters currently chosen being a valid prefix, which is decided by the isPrefix() method in the Dictionary class	1	2
Choose game mode	4.2	Daniyal	As a user, I would like to choose if I would like to play against a computer or another player sitting beside me	Before the game starts, ask the player what mode they want to play in with two buttons on the screen		
Player vs Player	4.3	Daniyal	As a user, I would like to be able to play against another user on the same grid, where the second player needs to find words that the other did not to score points	All code for player actions must be written In a way that allows more players to be added on later	1	3

SECTION 3: SOFTWARE DESIGN

Design pattern #1: Visitor pattern
When counting the score of each individual word a player may find, the word will be passed into a visitor. The visitor will then verify the eligibility of the word, and return a score for that word accordingly.

This pattern will be used to implement our user story 1.6 (Update Counting Score)



<< interface >>
Ivisitor
+visit(allword: ArrayList[String]): integer
+visit(word: String): integer
<< interface >>
Ivisitable
+accept(Ivisitor): integer
Wordvisitor
+visit(allword: ArrayList[String]): integer
+visit(word: String): integer
Words_found
+accept(Ivisitor): integer

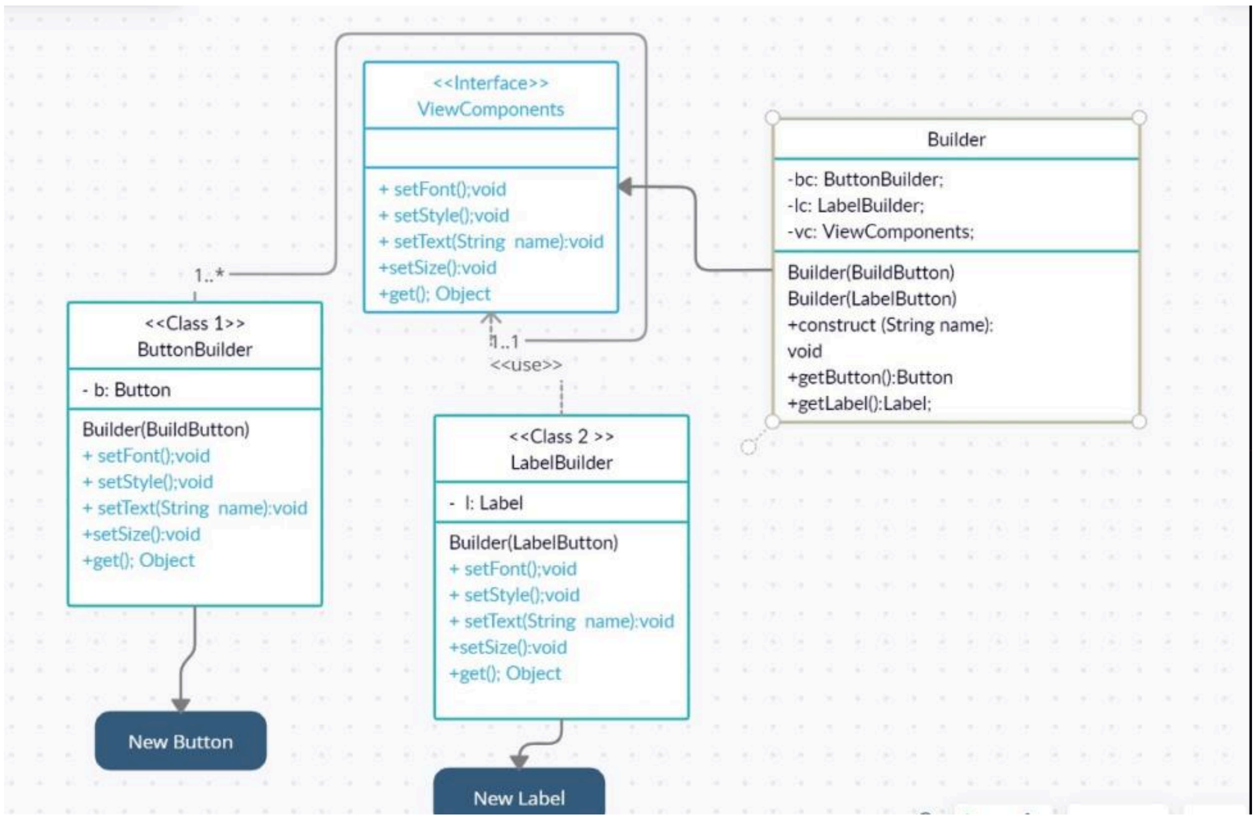
Implementation Details: The UML diagram outlines these main components:

- The Wordvisitor interface, which includes visit methods that are separated by inputs.
- The Words_found interface contains an accept method which uses a visitor to return some specific value when counting word or words.

To achieve this update, I will create two interfaces, Ivisitor and Ivisitable. While words_found implements Ivisitable, it calls Wordvisitor which implements Ivisitor when using the accept method for a visit. Wordvisitor will then determine which visit method it should use and returns back the valid output, then the value returned will be the result of running the accept method in Words_found visitable.

Design pattern #2: Builder

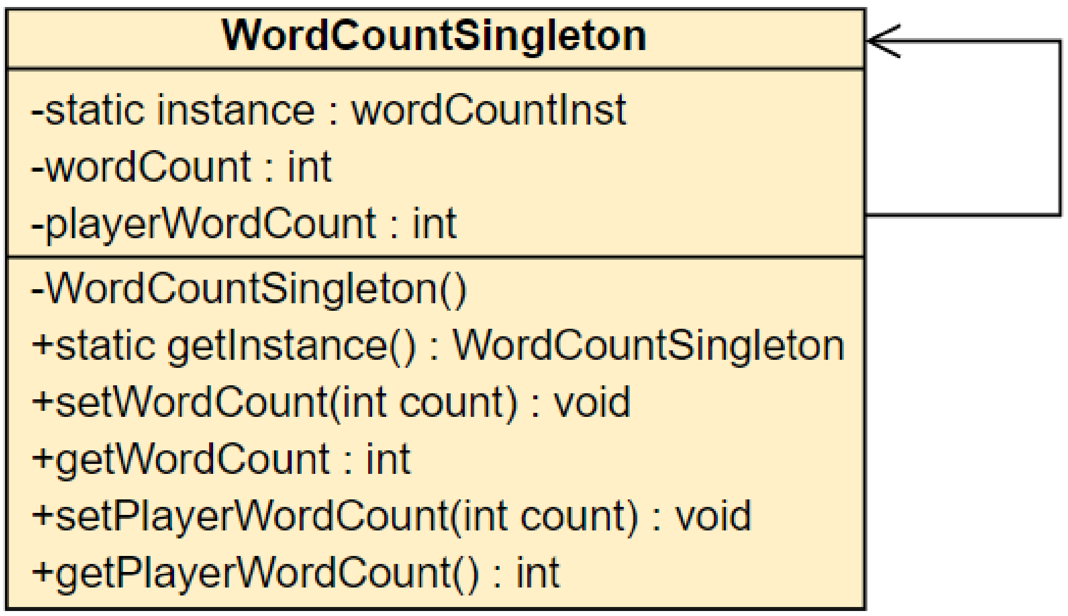
Overview: This pattern will be used to create the buttons and labels needed in some of the user stories, such as 2.8 (guess with buttons).



The *ButtonBuilder* and *LabelBuilder* class provide parameters to create a generalized button/label. After the new *Builder* is instantiated with a specific kind of builder, it can construct new components using the *construct* method, and then give them back to the class the builder is called by using the *get* method. The class is then free to alter any of the generalized parameters should it need to.

Design Pattern #3: Singleton

This pattern will be used to implement our user story 1.4 (Word count)


WordCountSingleton
-static instance : wordCountInst
-wordCount : int
-playerWordCount : int
-WordCountSingleton()
+static getInstance() : WordCountSingleton
+setWordCount(int count) : void
+getWordCount : int
+setPlayerWordCount(int count) : void
+getPlayerWordCount() : int

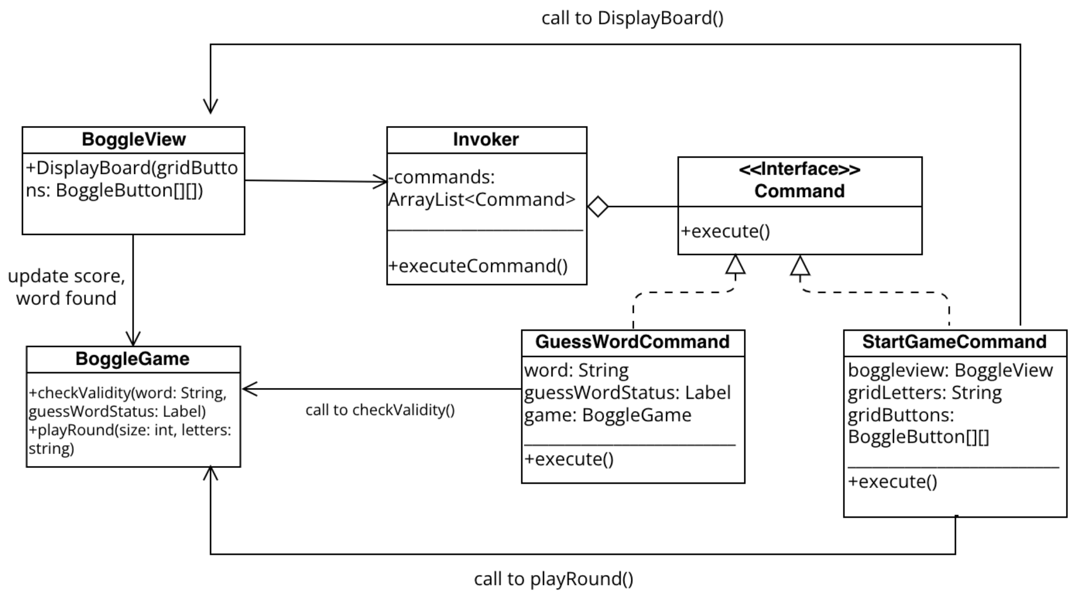
Implementation details: The UML diagram above outlines these main components:

· The *WordCountSingleton* class, which includes 1 class constructor "WordCountSingleton" and 3 methods "getInstance", "setWordCount", and "GetWordCount".

To facilitate the user story, Singleton design pattern is being implemented here. A new singleton class type *WordCountSingleton* is created containing a static instance of self type *wordCountInst*. A private integer *wordCount* is used to store the current value of current word count (remaining words to be guessed). A static method *getInstance*() is used to initialize an empty instance of the class itself. Two methods getter and setter are used to get and set the value of private integer *wordCount*. Two methods getter and setter for *playerWordCount* to see how many words are guessed by the player. Then we subtract *wordCount* by *playerWordCount* to update available words on the boggle game.

Design pattern #4: Command Pattern

Overview: This design pattern will be used to implement user story 2.2 (guess word button) and 3.2 (implement command pattern).



Implementation details:

- The Invoker class represents an Invoker object which receives commands from the player of the game through the JavaFX view. These commands are given by its command attribute, and it executes these commands using the method executeCommand(). The invoker has an aggregation of commands that it can execute.
- The Command interface represents a command from the view. This interface has the method execute(), which is meant to execute the command appropriately.
- GuessWordCommand is an implementation of the Command interface. It has a word parameter which is a string representing the word being guessed, a label parameter which is the status of the guess displayed on the screen, and a game parameter which is a reference to the BoggleGame object for this game. The execute() method calls the checkValidity() method in theBoggleGame class
- StartGameCommand implements Command as well, and contains a reference to the BoggleView object, and a string representing current grid letters, an array of BoggleButton objects which are the buttons on the screen. The execute method calls the playRound() method in the BoggleGame class.

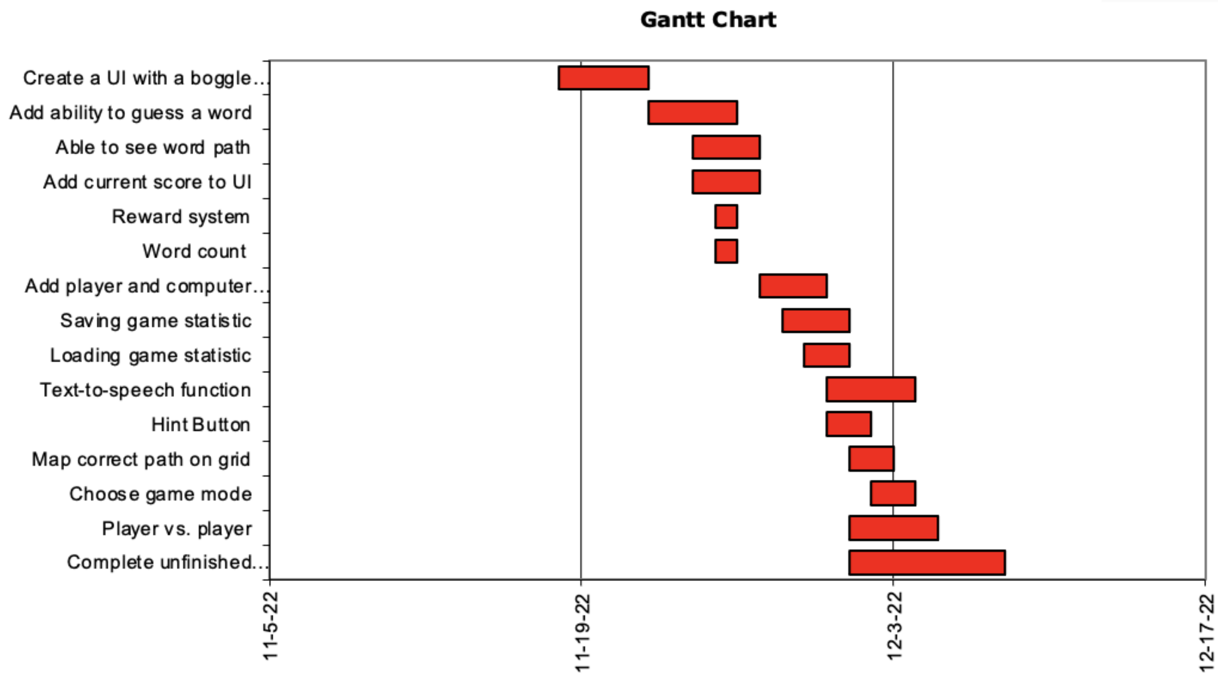
To implement user story 2.2 and 3.2, when the user presses the guess button on the UI, a command is added to the commands parameter in the invoker object, which is initialized in BoggleView. Then, the executeCommand() method can be called to execute the most recent command. Here, a GuessWordCommand is passed in, so the invoker calls its execute method and the GuessWordCommand object passes its parameters to the checkValidity() method in the BoggleGame class where score and word counts are adjusted. To implement starting a game, when the BoggleView creates and sends a StartGameCommand to its invoker object, then calls executeCommand() in the invoker, the execute method of StartGameCommand is called. This then calls playRound() in BoggleGame to start a round, and DisplayBoard() in BoggleView to start displaying the board.

SECTION 4: EXPECTED OVERALL PROJECT TIMELINE

- **What is the current team's assessment of the project timeline?**

  This project should take 4 weeks to complete
- **What are the major milestones?**
  - Getting the basic UI with the boggle grid working
  - Adding ability to click on letters to form a word
    - Pressing a "guess word" button and seeing if it's correct
  - Implement turns for player and computer
  - Adding saving of current score and reading save file for high score
  - Adding accessibility features such as narrator
  - Adding the ability for two players to play against each other

Sprint 1: Nov 18 - 25, **Sprint 2:** Nov 25 - Dec 2, **Sprint 3:** Dec 2 - Dec 7


Gantt Chart
Create a UI with a boggle...
Add ability to guess a word
Able to see word path
Add current score to UI
Reward system
Word count
Add player and computer...
Saving game statistic
Loading game statistic
Text-to-speech function
Hint Button
Map correct path on grid
Choose game mode
Player vs. player
Complete unfinished...