Design Document

Version 1.0 – 2022.11.13

Created 2022.11.05

Boggle

Douglas Quan
Christopher Marco
Suleiman Farah
Gunjot Ghuman

SECTION 1: PROJECT IDENTIFICATION

This section should include a concise statement that address such questions as:

- Why are you doing this project (i.e. what is the motivation?)
- How will it enhance or add to functionality that already exists?

Our development team - the TSDC - decided to tackle the Boggle game as the final group project for CSC207. We all have a mutual interest in game development and the Boggle game from assignment 1 had a noticeably large capacity for creative development/expansion. In assignment 1, we completed a functional Boggle program that facilitated input streams via the terminal to engage with the user. We hope to develop the project towards a GUI-based boggle game with the appropriate accessibility features attached. Visually, a player should be able to click on an 4x4 (or 5x5) button grid each housing a letter to piece together a word causing changes to the view and subsequent actions by the controller and the model. This feature essentially pushes assignment 1 to its completed form, an aesthetically pleasing GUI-based game where all interactions begin and end with the view to simulate a proper gaming experience.

SECTION 2: USER STORIES

Name	ID	Owner	Description	Implementation Details	Priority	Effort
GUI Screen	1.1	Suleiman	As a user, I want to be able to interact with the game in a visual capacity.	Create an aesthetically pleasing GUI screen with the appropriate scene elements (buttons, searchbar, textbox).	1	1
Timer	1.2	Douglas	As a developer, I want the game to be time limited so that the user will not be playing the game endlessly. It makes the game less boring.	Create a drop-down menu with different amounts of time for a user to choose at the beginning of the game. Once time is up, the game terminates.	1	2
Word progress feature	1.3	Suleiman	As a user, I want to be able to see the progress I made while guessing a word so I don't need to memorize what buttons I pressed.	Draw a line between all the buttons pressed in order and store the lines in an iterative collection.	1	3
Correct Words Notifications	1.4	Christopher	As a user, I want to know whether my inputted word is a real word or not by getting a simple notification.	Using a State Pattern that has 2 states of notifications. Each state will import JavaFX Library to provide correct words or incorrect words notifications.	1	1
Undo word progress	1.5	Suleiman	As a user, when a word is guessed wrong I want all the lines to disappear in preparation for my next guess.	Create an iterative collection of lines whenever a button is pressed and clear when a word is guessed.	1	3
Total word tracker	1.6	Gunjot	As a user, I want to know how many words I've found while I'm playing the game	Create a single instance of words founds, increment by one every time a word is found and display it on the screen	1	3
Text-to-Speech	2.1	Suleiman	As a user, I may have trouble seeing so I would like all instructions to be spoken to me to assist my gaming experience.	Incorporate all instructions into the eventhandler and use the help of a google-speech-api to speak out the text when appropriate.	1	1

Game Music	2.2	Suleiman	As a user, I want to be immersed into the game I am playing so music will enhance my experience interacting with the program.	Add a JavaFX library that allows the import of music, play it automatically when the game is launched.	2	2
Pause Music	2.3	Suleiman	As a user, I want to be able to mute the music if I don't like the selection or I am listening to my own external music.	Create a button at an appropriate location on the GUI and add an event such that the music stops playing when the button is pressed. (Possibly draw a line diagonal over the button when music is muted and erase when enabled)	3	2
Control sound level	2.4	Gunjot	As a user, being able to control the loudness of the music is important so I can listen to the music how I want to.	create a sliding button to control volume level or allow the user to type a number that represents their volume level wanted. This button would use the	2	4
Collapsing Grid (Gamemode 1)	3.1	Douglas	As a user, I would like to have different game modes to choose from, so that there is a random aspect in the game that makes it unpredictable and entertaining. A random grid is removed every minute elapsed.(up to 10 grids?)	Create a button to activate the gamemode before the game starts. Get the time elapsed from the timer class. Every minute passed, randomly set a grid to the symbol "#", representing an unusable grid.	3	3
Three Swaps (Gamemode 2)	3.2	Douglas	As a user, I would like to have different game modes to choose from to make the game more entertaining. The user is given 3 chances to swap positions of 2 grids. This adds a strategic aspect to the game, since the user has to think how to utilize the 3 chances to optimize the score.	Create a button to activate the gamemode before the game starts. The user swaps the grids by clicking the 2 target grids to be swapped. Swap the position of the 2 grids in Position class.	3	2

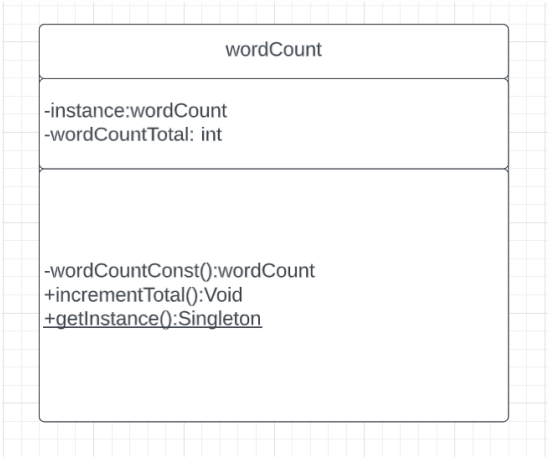
Randomize grid	3.3		As a user, if I don't like the current arrangement of letters on the grid I should be able to refresh it without ending my current round	Create a button that when pressed will call on the grid creator to create a new grid and put it on the gui	1	2
Game Level	4.1	Christopher	As a user, I want to know how capable I am to master the Boggle Game	Create an iterative collection of game levels ranging from the lowest number of words in the grid to the highest amount of words in the grid	2	1
Hint	4.2	Gunjot	As a user, I want to get a hint if I'm struggling to find a word after a long period of time	Allow the user to opt into using hints which will reveal the starting points on the grid for a word. They will opt in by pressing a button before the game beings	1	1
Dictionary	4.3	Douglas	As a user, I want to know the meaning of a specified word, so that I can learn a new word when I get a word accidentally without knowing the meaning of the word.	Create a button for the user to choose to look up the meaning of the word found. Use a dictionary api to fetch the meaning of the word. https://dictionaryapi.dev/	4	3
Increments in the Timer	4.4	Christopher	As a user, I want to play as long as possible during the timed gamemode. Therefore, adding 5 seconds for each correct word to the timer would provide a better BoggleGame experience.	Incorporate the word checker function with the timer in order to add another 5 seconds for each correct answer.	3	2
Limit Computer	4.5	Gunjot	As a user, I want to limit how many words the computer is allowed to find	Instead of looking for all words in the boggle grid, stop the program after a certain amount of words have been found	1	4

SECTION 3: SOFTWARE DESIGN

Design Pattern #1: Singleton

Overview: This pattern will be used to implement User Story 1.6 (total word tracker)

UML Diagram:



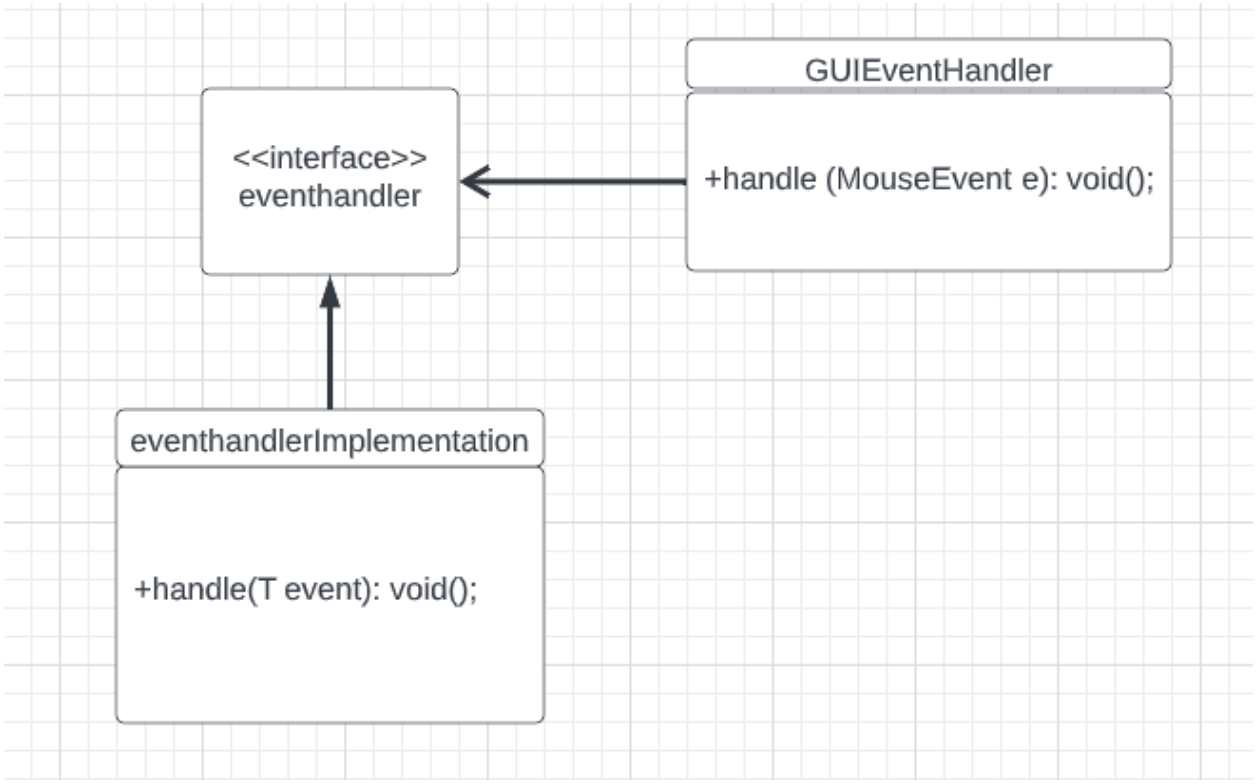
Implementation Details: The UML diagram outlines these main components:

- the wordCount singleton which is the class that will be used across the program to track the amount of correct words that have been guessed
    - The instance and the constructor of wordCount are private, so the only way to access and create a wordCount instance would be to use getInstance() which will create an instance using wordCountConst if none exists yet
    - The incrementTotal will be the method that increments wordCountTotal each time, which is a private variable

Design Pattern #2: Observer Pattern

Overview: This pattern is mainly implemented in conjunction with the GUI screen user story. All elements of the GUI scene are considered observables and any changes applied (button click) will be met with a specific response once observed.

UML Diagram:



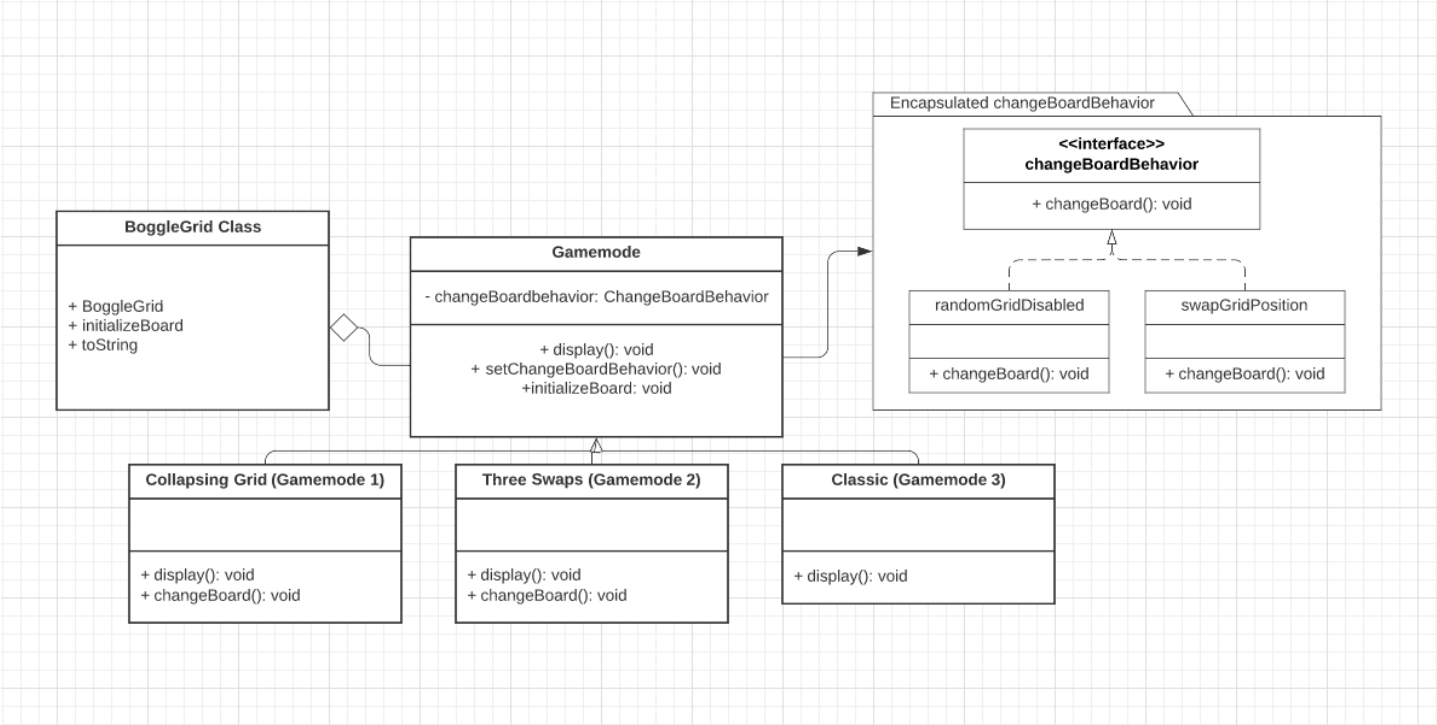
Implementation Details: The UML diagram focuses on a few key components

- The eventhandler interface houses one method called handle which expects a type of interaction with the program and an event of that type (ie. a mouse click and a mouse click event).
- A class set aside to specifically handle GUI events, the GUIEventHandler implements the eventhandler interface along with the sole handle class.

Design Pattern #3: Strategy Pattern

Overview: This pattern will be used to implement User Story 3.1 and 3.2(Gamemodes). The way how the board changes varies from different gamemodes, specifically disabling a grid randomly over time in gamemode 1 and swapping grids specified by the player in gamemode 2.

UML Diagram:



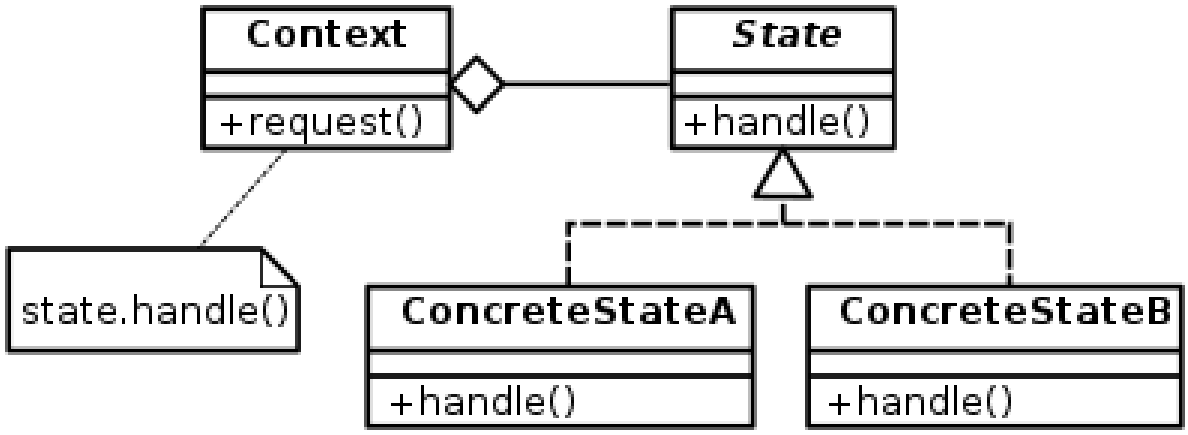
Implementation Details: The UML diagram outlines these main components:

- *BoggleGrid class:* requires changeBoardBehavior
- *Encapsulated changeBoardBehavior interface*: encapsulates the variation in the change of board and separates them from the other parts that stays the same among gamemodes.
- *Three gamemode classes*: gamemode 1 and 2 possess variation in changeBoardBehavior, while gamemode 3 remains the same
- *randomGridDisabled class*: gets the time elapsed from the *Timer class*. Every minute passed, randomly set a grid to the symbol “#”, representing an unusable grid, to disable the grid.
- *swapGridPosition class*: swaps the position of the 2 grids selected by swapping the position in *Position class*.

Design Pattern #4: State Pattern

Overview: This pattern will be used to implement User Story 4.1(Words Input Notifications). Through this pattern, the BoggleGrid will always provide a notification to the User's inputted word, whether it is a correct input's notification or an incorrect input's notification in the form of simple sound from JavaFX Library.

UML Diagram:

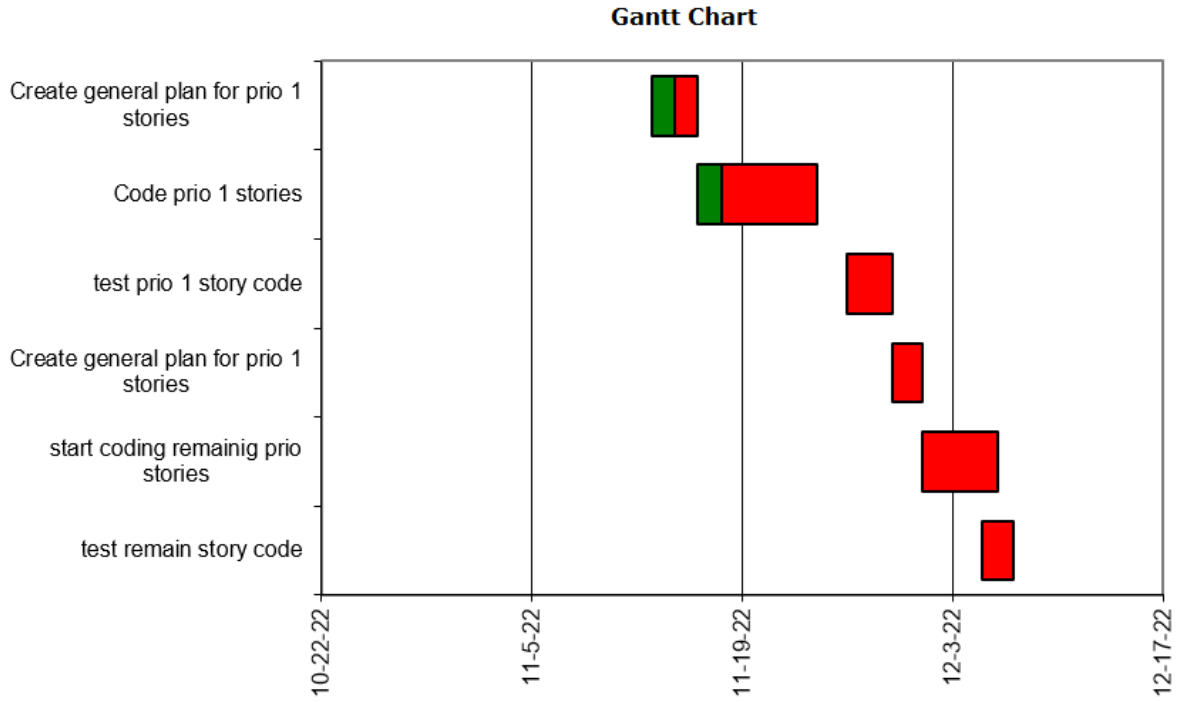


Implementation Details: The UML Diagram points out 4 main elements:

- The handle function which will send
- The context that has a request function
- The state that includes 2 ConcreteState: Correct Words and Incorrect Words.

The handle method of an Object requires an input of word that need to be checked. Every sent word will request a state the word is in, if it is classified as a State of Correct word, it will give the user a Notification of Correct word and vice versa.

SECTION 4: EXPECTED OVERALL PROJECT TIMELINE



Task	Start Date	# Days Required	Percent Complete
Create general plan for prio 1 s	11-13-22	3	50
Code prio 1 stories	11-16-22	8	20
test prio 1 story code	11-26-22	3	0
Create general plan for prio 1 s	11-29-22	2	0
start coding remainig prio stori	12-1-22	5	0
test remain story code	12-5-22	2	0