SYSC 3110 Milestone 2 Documentation
Group 17:
Oyindamola Taiwo-Olupeka - 101155729
Oluwatomisin Ajayi - 101189490
Edidiong Okon - 101204818
Ese Iyamu - 101081699

TABLE OF CONTENT

Section	Content	Completed By
A	Design Decisions	Oyindamola Taiwo-Olupeka
B	UML diagrams	Oluwatomisin Ajayi
C	User Manual	Ese Iyamu
D	ReadMe	Edidiong Okon

A. Design Decisions

For the ScrabbleGame project, we collectively decided on seven(6) classes and one(1) interface- Scrabblemodel, ScrabbleBoardFrame, Player, ScrableStart, MainFrame, ScrabbleController are classes and ScrabbleView is an interface. We implemented the MVC pattern in this milestone.
The model was the contents of the game, it notifies the view when any change has been made. The is the graphical representation of the model, it displays the look of the content and subscribes to the model to ensure that its appearance reflects the current state of the board model. Lastly, the controller handles the input, therefore setting the behaviour of the input. It updates the model according to the input from the user and the updates are observed by the view.
This is subject to change as the project progresses.
In a traditional Scrabble game, 2-4 players randomly choose 7 tiles to place on their tile rack. During their turn, they pick a combination of tiles and play on the squares on the board.

Breakdown of classes:

1. **ScrabbleModel:**
   This class initializes the scrabble game. It generally controls the entire flow of the game by distributing 7 random Tiles to each player, verifying the legality of moves and holding all vital information. It keeps track of the 2-4 players (specifically, the order they play in, their score, and the start and end of their turns). And lastly, it determines the winner at the end of the game by comparing all the players' scores.
   The play method asks for the number of players and then enters a loop that plays words (horizontally or vertically) on the specified squares. The game loops through the players until a player has no more tiles, and the game is over. ScrabbleGame keeps track of each player's score and determines the winner by whose score is the highest.
   It makes use of ScrabbleWords.txt file with a buffered Reader and verifies the legality of every played word in the Scrabble game.

2. **ScrabbleBoardFrame:**
   This class initializes a scrabble board frame and is basically in charge of the GUI of the board. The board is up of a 15x15 array of characters.
   This class has 4 nested classes. We decided to use nested panel classes to make our work organized.
   These classes are
   - BoardPanel - which initializes the board which holds the grid layout for the traditional scrabble board.
   - RackPanel- rack panel that represents the rack for every player. With each turn, the label on the panel changes per player move.
   - ScorePanel- Store the player's name and their scores. It also has information about the values associated with the tiles for user understanding.
   - ButtonPanel- initializes all the buttons for the game. Such as play, clear, pass, and end game.

3. **Player:**
   This class represents the players in the ScrabbleGame. It holds the name, score, and tiles It is a dependent class that provides information for the model and frame.

4. **ScrabbleStart:**
   Starting screen of the game. Asks players to input their names and initializes some buttons.

5. **ScrabbleController:**
   Observes the user inputs using an action listener and mouse listener and performs the actions based on user input.

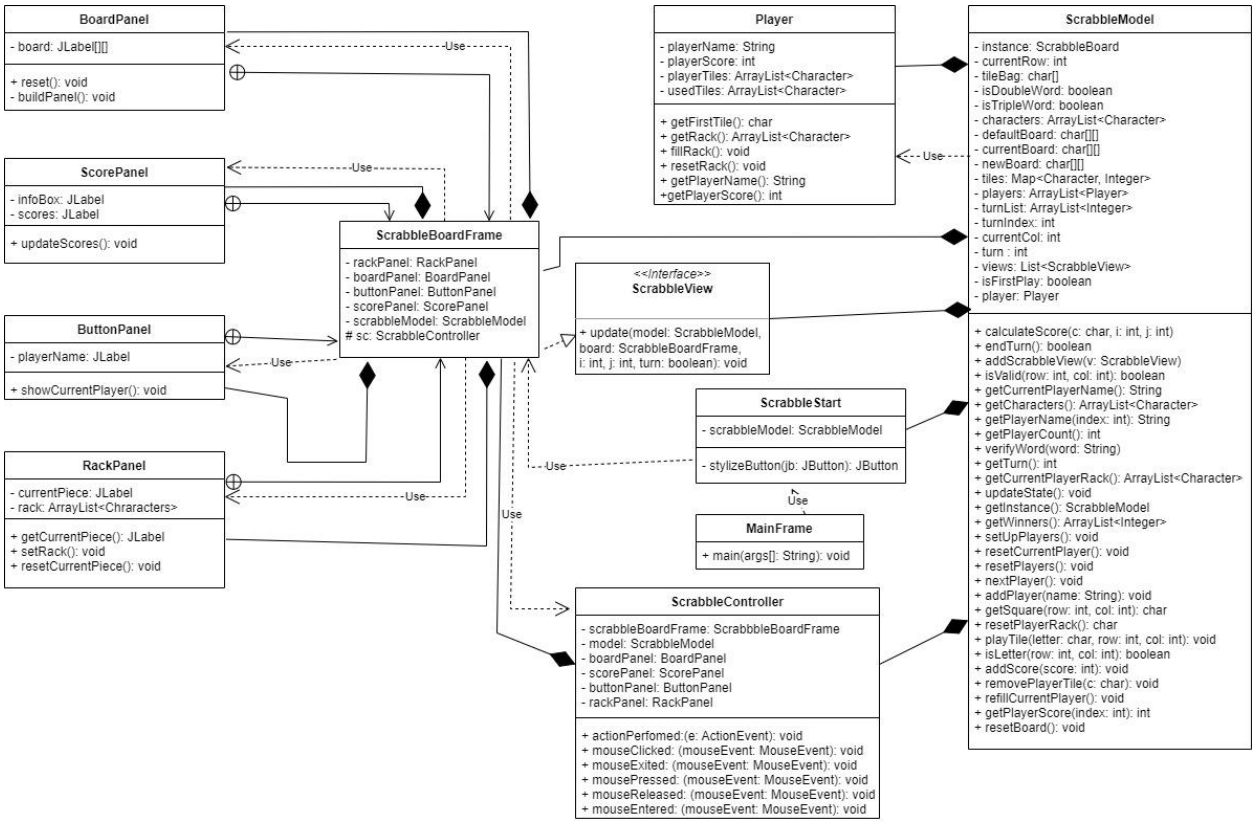
6. **ScrabbleView:**
   Manages the views of the game.

7. **MainFrame:**
   The main class that starts the game. It calls ScrabbleStart first for stylistic choices.

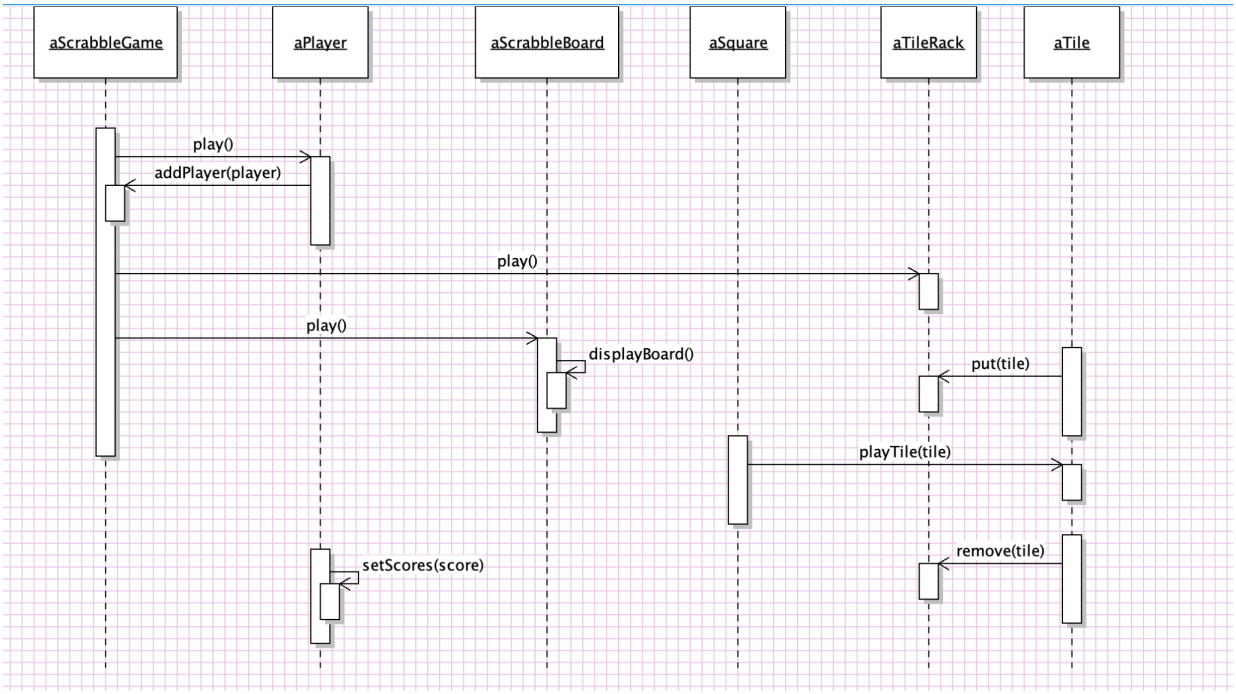
To conclude, the classes all have individual responsibilities but they work together to make ScrabbleGame functional. All classes are subject to change at each milestone.

B. UML Diagrams

Class Diagram



Sequence Diagram



- Games loads.
- Players are added to the game.
- Tile rack is created and a scrabble board is displayed.
- Players start playing. Once a tile is played, it is removed from the rack.
- A winner emerges based on the final score.
- Game ends.

C. User Manual

To Play the board game Scrabble:

- Click on the jar file to start the game.
- Enter the names of the players. If no names are entered players would be assigned names.
- Click start game
- Players are able to use the following buttons for game play: - “clear”, “play”, “ pass”, “End Game”.
  Clear:- This button is used to clear the letter tiles placed by the player on the board. This button can only be used before Play is pressed.
  Play: - This button is used to “play” the letters placed by the Player. Once this button has been pressed the players' played letters are locked and cannot be changed.
  Pass: - This button is used by the players to pass their turn to the next player.
  End Game: - This button is used to end the game.

- The first player starts by clicking the letter they want on the board from their rack and then clicking on the space* they want it on the board.
- Players are allowed to place their next letters only next to letters tiles already on the board. Words can be formed horizontally or vertically only from existing letters.
- Player's score is calculated and given back as many tiles as they used to play to replace their rack. E.g say a player's rack was [A, F, H, A , G, T , U] and they played letters "H, U, T". The player's rack is filled with 3 letters and their rack would be ["A, F, D, A, G, F, B].
- All new words(horizontally or vertically) formed by the auditions of new tiles must be comprehensible. Tiles can't be moved around once they've been played and scored.
- Up to two blank tiles can be used**.
- Game ends when all letters have been drawn or when players have used up all their tiles or when all possible moves have been made.

Note:
* The first player has to put their first letter at the middle of the board.
** Player would have to state what letters the blank represents and it remains that letter till the end of the game.

D. ReadMe

Scrabble
A simplified version of the classic word game, Scrabble, built using Java and java.util packages.

Milestone 1
A text-based version of Scrabble where players use a keyboard to play the game via a console.

Authors
Ese Iyamu
- Created the Player and Controller class.
- Worked on the user manual aspect of documentation.
- Added Java Doc comments to classes and methods.

Oyindamola Taiwo-Olupeka
- Worked on the ScrabbleBoardFrame and MainFrame classes.
- Wrote the design decision document.
- Added Java Doc comments to classes and methods.

Oluwatomisin Ajayi
- Worked on the ScrabbleModel class and ScrabbleView interface.
- Created UML class and sequence diagrams.

- Added Java Doc comments to classes and methods.

Edidiong Okon

- Worked on the ScrabbleStart classes and the test cases.
- Wrote the README file.
- Added Java Doc comments to classes and methods.

Known Issues

- The play, clear, pass, and end game buttons are not active currently. We believe the issue is with the controller class. We aim to fix it completely for the next milestone.

Future Milestones and Deliverables

Milestone 2

- GUI-based version of the Scrabble game.
- Add a GUI to the current text-based version of the game using GameView and GameController classes.
- Create unit tests for the Model.
- Update all UML Diagrams and documentation.

Milestone 3

- Add new features and support for blank tiles, premium squares and AI players.
- Add a method that returns all the possible legal moves and plays the highest scoring move.
- Refine the program to make it robust and smell-free.
- Update UML Diagrams and documentation accordingly.

Milestone 4

- Add a multiple-level undo and redo feature.
- Implement Java Serialization and utilize it to add a save and load feature.
- Create custom boards with the alternate placement of premium squares using XML or JSON.
- Update documentation.

Change Log

- We removed the Tile and Tile rack class and imputed them into the Model class because they did not have too many responsibilities.
- We made the ScrabbleGame the the Model class
- We made ScrabbleBoard the frame class
- We also added the controller and view classes

Roadmap

November 11, 2022

- GUI (View and Controller).
- Unit Testing for the Model.

November 21, 2022

- Creation of blank tiles, premium squares and AI players.
- Design of Scrabble move generation algorithm/method.

December 5, 2022

- Addition of multiple game level undo and redo features.
- Save and load features.
- Creation of custom boards.