

SYSC 3110 Design Decisions

Milestone 2

Group 8

Group Members: Sandy Alzabadani, Elyssa Grant, Gillian O'Connell, John Khalife

Author: Elyssa Grant, 101258660

Board

For Board, to create more descriptive errors, Board was given an `ErrorEvent` instance variable, `status`. This likewise created the need for a `getStatus()` method so that any errors that the Board generated could be accessed and presented to the player. Features that helped create a text-based appearance such as the `toString()` method were removed.

Display

Since our project no longer uses text-based input and output, `Display` was no longer a useful class. Thus, it was deleted entirely from our project and was replaced with the `ScrabbleView` class.

ErrorEvent

`ErrorEvent` is a new class that is used to provide more descriptive information about any errors that occur in the `Board` class. `ErrorEvent` contains an enumeration with all the different error types, as well as methods to set and obtain the error stored inside.

Game

Since the `Scrabble` game is now in MVC pattern, the main function was deleted entirely from `Game`. To accommodate the MVC pattern, `Game` was then given an `ArrayList` of `GameObservers`, as well as methods to add and remove the views observing the `Game`. To better accommodate the MVC, a new variable was added, `currentPlayer`, which stores the index associated with the player whose turn it currently is. Additionally, a new method called `handleNewTurn()` was added in order to handle the logistics of changing which player's turn it is in the `Game` class.

GameController

The `GameController` class is the Controller portion of the MVC pattern. This class is responsible for parsing the user's inputs based on the button pressed and calling the appropriate methods in Model to execute the corresponding commands. The class shares an instance of `Game` with `ScrabbleView`, so that the Model-View-Controller setup is preserved, and so that the Controller can access the appropriate methods in the model to call.

GameObserver

`GameObserver` is an interface developed to decrease the coupling between the Model and the View in the MVC plan. It has general methods for updating based on the board and scores changing, and it is up to the View to implement these as deemed necessary.

LetterBag

In order to make unit testing easier, a static method called `emptyBag()` was added to the class. This method removes all the letters from the bag, allowing easy resets when testing the methods in this class, as well as other classes that rely on it.

Player

Player was originally a large source of I/O, which is not conducive to the MVC pattern. As such, it required some reworks. The first thing that was done was splitting the `playerTurn()` method into the main `playerTurn()` method, and three smaller classes. One of these new classes is `placeLetter()`, which notes the letters that have been designated by the player to be used. Alongside it is `addCoordinate()`, which likewise notes the coordinates that the player wants to add letters to. `exchangeLetters()` acts as a helper function the `playerTurn()`, adding more cohesion to the `playerTurn()` removing its secondary function of exchanging letters with the bag and also developing the word/location combination to add onto the board. `updateScore()` had some changes as well, notably now returning a boolean that indicates whether the game has finished or not. Since the game is no longer text-based, Player's scanner instance variable was also removed.

ScrabbleView

ScrabbleView is the View segment of the implemented MVC pattern. ScrabbleView's main functionality is to provide the visual appearance of the game, and to update the visuals to reflect changes in the Model. To this end, ScrabbleView inherits from `JFrame` to be able to add GUI components to it and implements `GameObserver` to interact with `GameObserver` in a loosely coupled way. ScrabbleView contains all the modifiable GUI components as instance variables for easy access to them. It also contains a copy of `Game`, so that it can set up the MVC pattern when its constructor is first called. ScrabbleView implements all the methods in `GameObserver`, as well as methods to properly display the board, scores, and the rack. Finally, ScrabbleView contains the main function, as per the standard in an MVC model, though it simply calls the ScrabbleView constructor, which handles the rest of the GUI setup.