Project 6 Update

Multiplayer Scrabble Game
Alex Walker & Jack McKinstry
Repository: https://github.com/alwa2786/OOAD_Project_5

Status Summary

Work Done:

Alex:

- Designed Model classes (Board, Tile, Player, etc.) to reflect the state of the game
- Created basic MVC implementation with GameView showing state of the board from the model
- Implemented Board and TileBag factories to populate the game board with bonus squares and traditional distribution of letter tiles
- Implemented MoveTile command to move Tiles within the Board and Player's inventory
- Refactored FinishTurn algorithm to improve functionality of verifying legal placement of Tiles

Jack:

- Added support for cycling through players, displaying Tile inventory based on the current active player
- Added additional views for Menu and Leaderboard screens with ability to select number of players and view final scores
- Implemented StartGame command to initialize players
- Implemented FinishTurn command to check/finalize Tile placement and increment player scores

Changes or Issues Encountered:

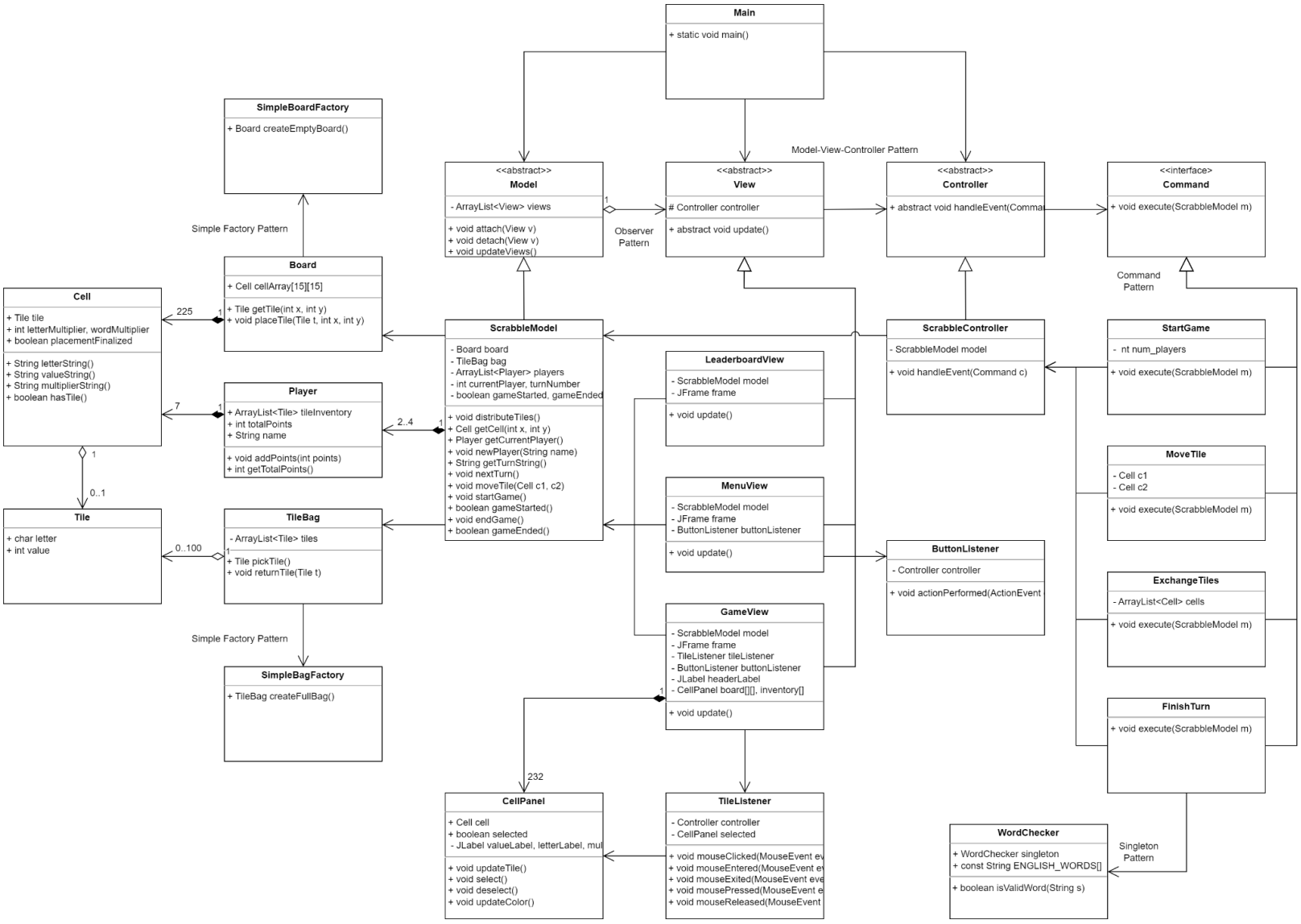
- Multiple views were added (MenuView, LeaderboardView) in addition to the originally planned ScrabbleView to enable ease of switching between the several views depending on the state of the game and improve cohesion of classes.
- Variables to identify the current state of the game (started/ended) and aid in determining which view should be displayed were added.
- A Listener to enable the behavior of buttons in the MenuView and ScrabbleView was added in addition to the TileListener that enables functionality of moving Tiles.
- The PlaceTile and RemoveTile commands were simplified into a single MoveTile command that ensures the legality of the action.
- Significantly more functionality was delegated to the ScrabbleModel class than was outlined in the original UML diagram.

Patterns:

- The Model-View-Controller pattern is implemented with abstractions of each and concrete ScrabbleModel, ScrabbleView (and other views), and ScrabbleController classes completing their respective behaviors.

- As part of MVC, The Observer pattern exists in a Subject/Observer relationship between the Model and the View.
- The Command pattern is implemented for the StartGame, FinishTurn, MoveTile, and ExchangeTiles commands with the ScrabbleController class functioning as the invoker and the ScrabbleModel class functioning as the receiver.
- The Singleton pattern (with lazy instantiation) is implemented for the WordChecker class, but the word verification is not yet fully functional.
- The Simple Factory pattern is implemented for the population of a TileBag object to be filled with the necessary number of Tiles (with their specific properties) and for the creation of a Board with the proper cells containing word/letter bonuses.

Updated Class Diagram



Plan for Next Iteration

The majority of main game functionality has been implemented in the first iteration of work. The game is in a playable state with most of the main mechanics complete (though fair gameplay in the project's current state would require players to verify the validity of words themselves).

Design changes that we have made to the application will have negligible impact on the timeline of features we still intend to develop for the complete application.

The features and functionalities that we plan to implement in the final iteration include:

- Exchanging Tiles that a player holds in their inventory for other Tiles in the TileBag.
- Verifying the validity of words placed by a player. The application currently does not check if the words played are in the scrabble dictionary (and thus legal to be played).
- Verifying that all letters in a word played during a single turn are within either a single row or column rather than spread across multiple.
- Scores are currently calculated using only the sum of all newly-placed tiles, with applicable multipliers. However, the scoring system should also take into account previously-placed tiles that are a part of any newly-spelled words (and double-count if multiple words are spelled).
- Functionality of blank tiles. The TileBag contains 2 tiles which when in a player's inventory do not correspond to any specific letter and act as a "free space". When placed on the board, players should be prompted to select which letter they would like the tile to represent.
- Players' ability to set custom names rather than just being referred to numerically.
- Addition to ScrabbleView to allow the Leaderboard of all players' scores to be visible during gameplay (rather than only during their own turn).
- Aesthetic UI improvements primarily to the Menu for selecting players and the Leaderboard view at the end of the game.