The Three Musketeers: The Six Design Patterns

SeungA Jung 1003634999 jungseu3

Shawn Youssef 1007257040 youss144

Ataol Baran Oray 1004745109 orayatao

Yusuf Emre Kenaroglu 1007318751 kenarogl

University of Toronto, Mississauga

November 29, 2021

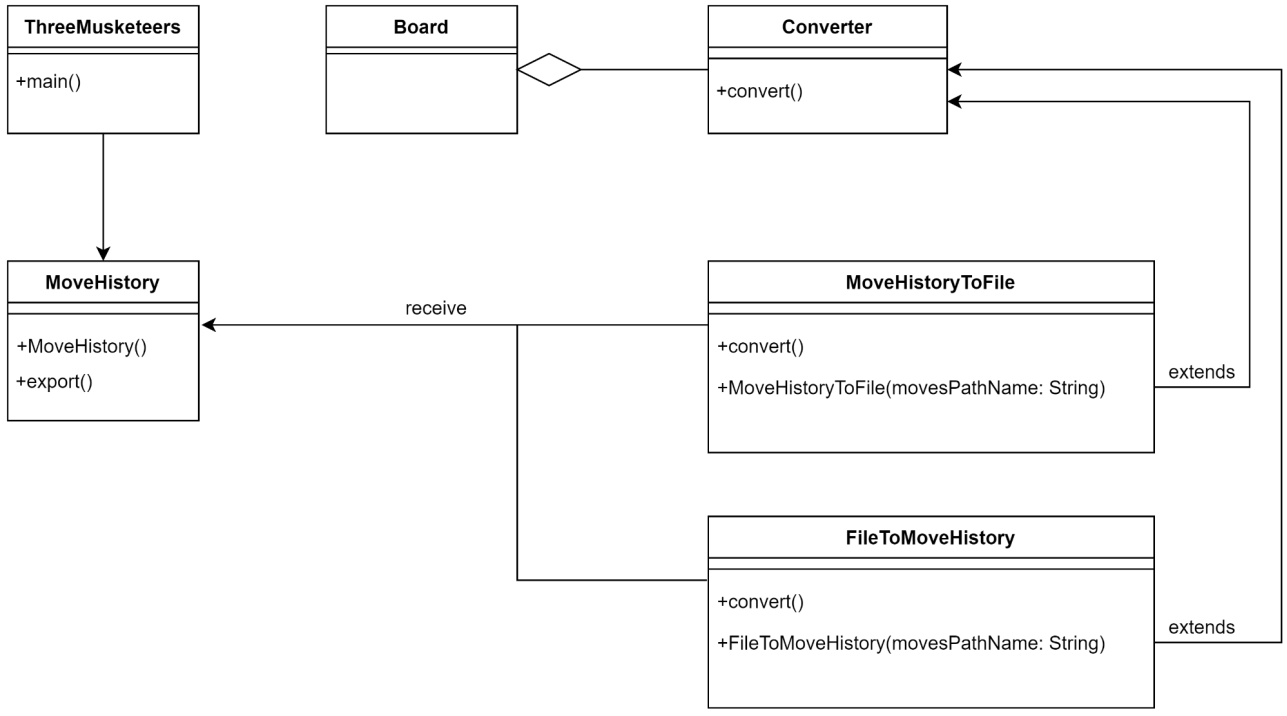
1. Command

Feature Name: Save Moves

Description

This pattern will convert a file into a MoveHistory class, and a MoveHistory class into a file using a command pattern to issue commands. This will save moves in separate files so that anytime a board is loaded the player will be able to undo any moves previously played. The command pattern will be used to differentiate between converting a file to a list of moves, and a list of moves to a file.

Command UML



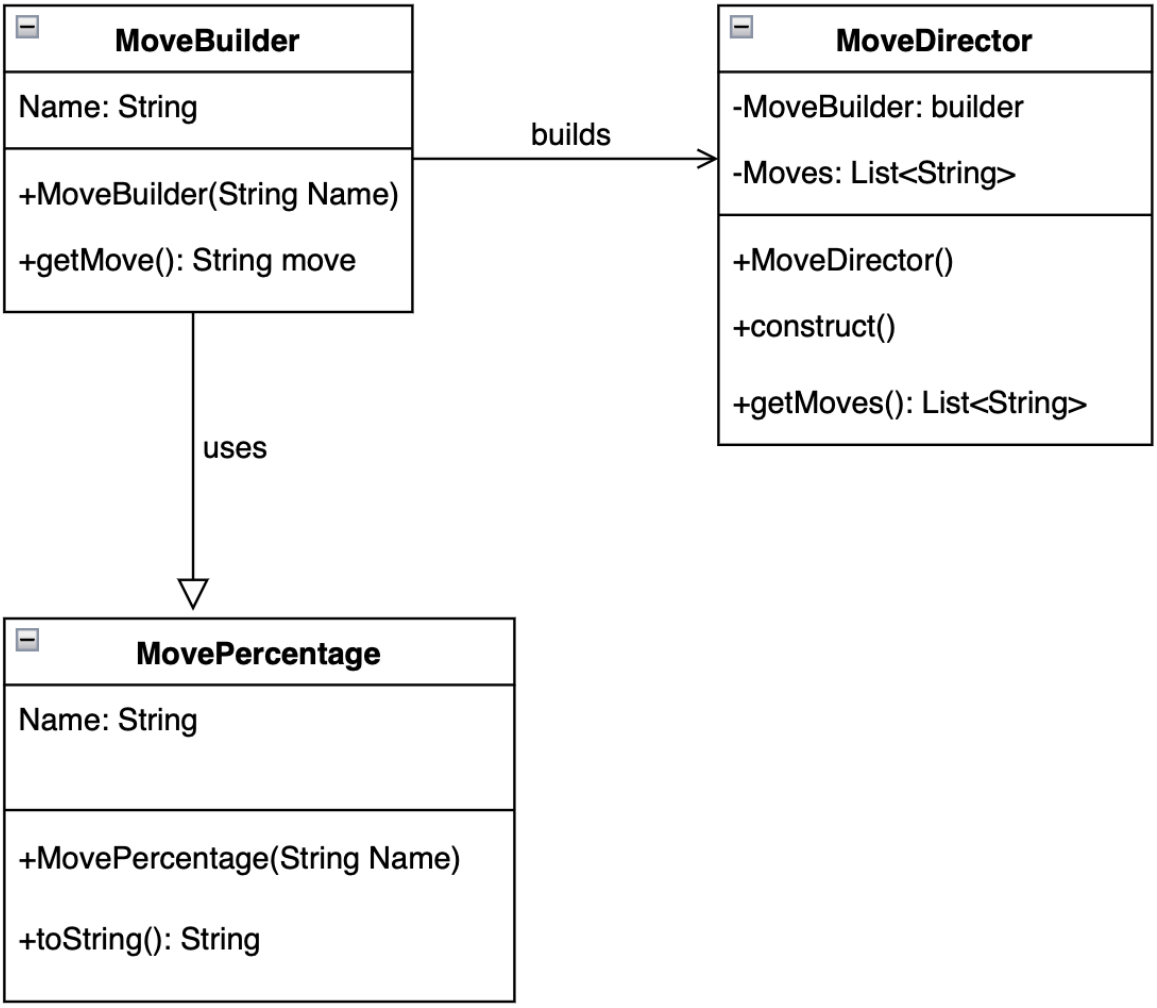
2. Builder

Feature Name: Chance Bar

Description

This feature adds an option for human player to play with moves being provided by the game coupled with each move's corresponding chance of winning the game. This option is for players who want to strategize their moves for optimal succession.

Builder UML



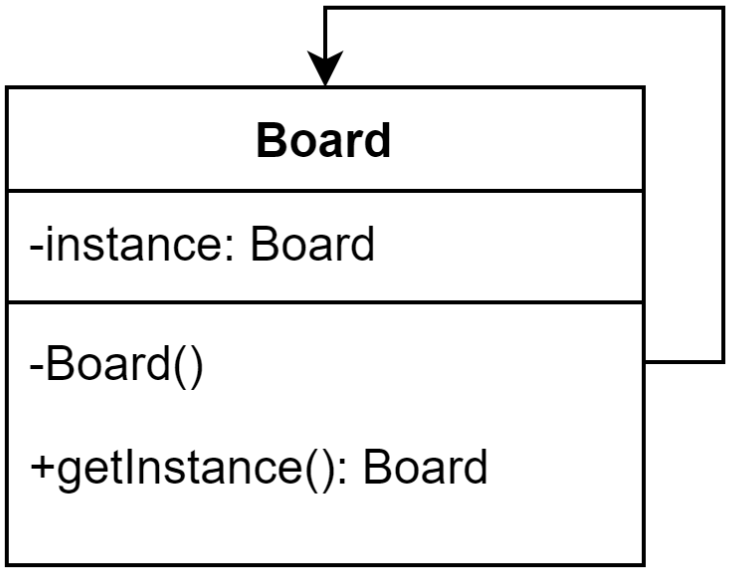
3. Singleton

Feature Name: Restart

Description

This allows the user to restart the game without quitting and/or suspending the current game. Using this pattern the current game is re-initialized rather than a new one being created.

Singleton UML



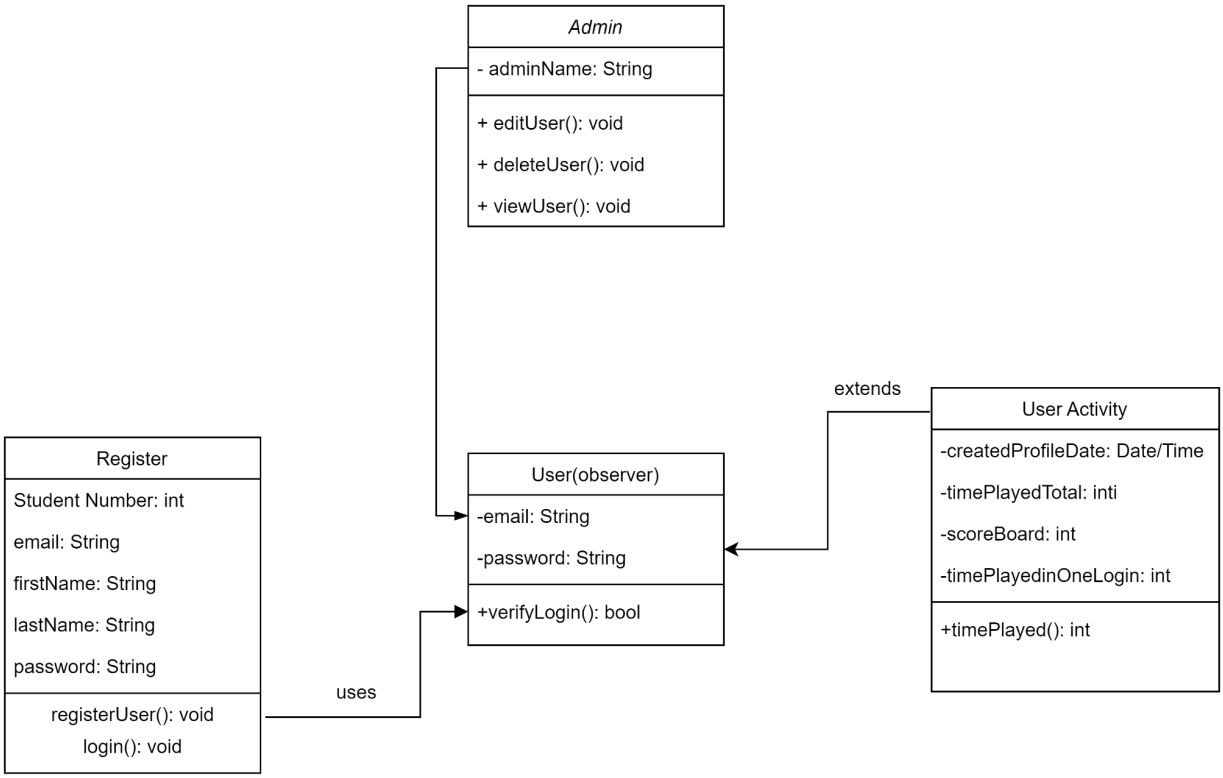
4. Observer

Feature Name: Registar

Description

This will allow users to store their own unique data and will provide a better user experience. It will store inputs from the new user and create a new profile according to it. This pattern then perfectly suits any process where data arrives from some input that is not available to the CPU at startup. This will also include a scoreboard of user ranks based on the total time played to win and number of moves the user made.

Observer UML



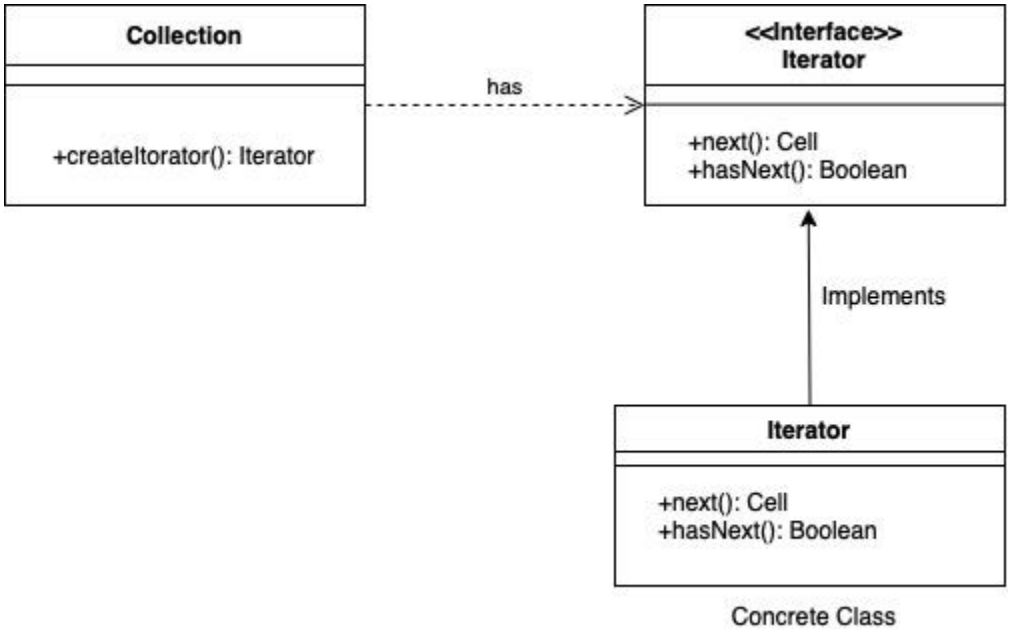
5. Iterator

Feature Name: Danger Alert

Description

Danger Alert feature is to alert the user that two Musketeer pieces are all in a column or in a row and that the next Musketeer move can lead to the guard winning. The feature will be alerted by changing the color of Musketeer pieces to red. Iterator pattern will be used to implement this feature. The iterator pattern will help to see if there is a next cell available for the Musketeer piece to move from possibleMoves().

Iterator UML



6. Adapter

Feature Name: Sound effects and interactions

Description

By using this pattern I want to encapsulate the interactions between sets of objects, by doing so I want to control their interaction independently to be able to use the sound effects accordingly. I used this pattern because there will be different types of media files which will need to adapt to related objects.

Mediator UML