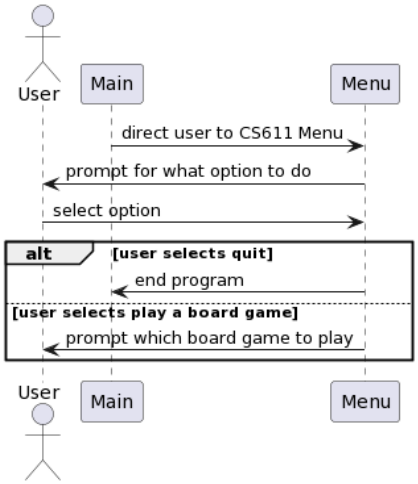
CS611 Assignment 3 Design Document

This document covers the high-level and compactly written object oriented design of ours, Reshab Chhabra and Gautum, CS611 Assignment 3. *Note: We are aware more classes can be made to abstract more content, but as we apply bias for action, we will add them once they are fully needed (SOLID).*

Project Architecture

UML Diagram for all my classes. This is also located in the Gradescope submission as "`Assignment3UMLDiagram.png`" in case this is too small



To begin, the code is initialized with the behavior interaction between Main.java and Menu.java. Then, once the user has decided on a board game to play, the menu will start() that board game.

This design structure to initialize a board game has been used since Assignment 1 and did not need to be adjusted.

Overall Organization Pattern

Key changes in addition to my assignment 2 are demonstrated in the following file additions:

```
•  BarSymbol.java             (part of Symbolable)
•  DashSymbol.java            (part of Symbolable)
•  Direction.java             (an Enum e.g. UP, LEFT, DOWN, RIGHT)
•  HorizontalWallSymbol.java  (part of Symbolable)
•  Inventory.java             (used in a player, later will be generic!)
•  PlusSymbol.java            (part of Symbolable)
•  QuoridorBoard.java         (implements Board but used in Quoridor)
•  TurnBasedGame.java         (will eventually be used in turn-based games)
•  VerticalWallSymbol.java    (part of Symbolable)
•  WallSymbol.java            (part of Symbolable, Packable)
•  Packable.java               (interface that says what goes in Inventory)
```

Originally, it was assumed in TicTacToe, Order of Chaos, SuperTicTacToe Restricted, and SuperTicTacToe Unrestricted that the board would automatically print the borders around the cells. However, when a came such as Quoridor comes here, we notice that the borders are actually part of the board and gameplay itself when placing walls. This inspired the design decision to actually make the QuoridorBoard class – a step towards a very generic RectangleBoard in the future: each content in the Cell[][] of the board can now include the border symbols itself, including PlusSymbol, BarSymbol, and DashSymbol.

Moreover, a new move feature was added: Direction. when it comes to moving in Quoridor – for both walls and the pawn — direction of the piece is considered. As it would be weird to type the exact index of where to move the pawn and it would be difficult to instruct placing down a wall, this enum class really makes it handy to prompt the user to place down a piece based on the direction.

A pattern that continued throughout is keeping the abstract board class and for each BoardGame adding its own type of Board — I believe this is the right direction too because when you think about it, every board game has its own customized version of a board (especially Catan for example!).

For previous design changes reference, please see "*CS611 Assignment 2 Design Document.pdf*" for more information.

Desired Next Steps

One thing I want to try later on is adding interfaces for certain types of games, such as one where we know there are Turns like the added `TurnBasedGame` class. I am keeping in mind not to keep things unnecessarily abstract since that is bad practice, but categorizing certain games can be useful if many are to come later in the semester.

Another task I'd like to embark on is making the jagged board toString: doing this would allow me to print any Board's toString and no override needed, which would be really cool!

There's more things I would like to try, but as I am currently still going to the hospital often and dealing with a broken foot and ligament, I am keeping my physical health in mind.