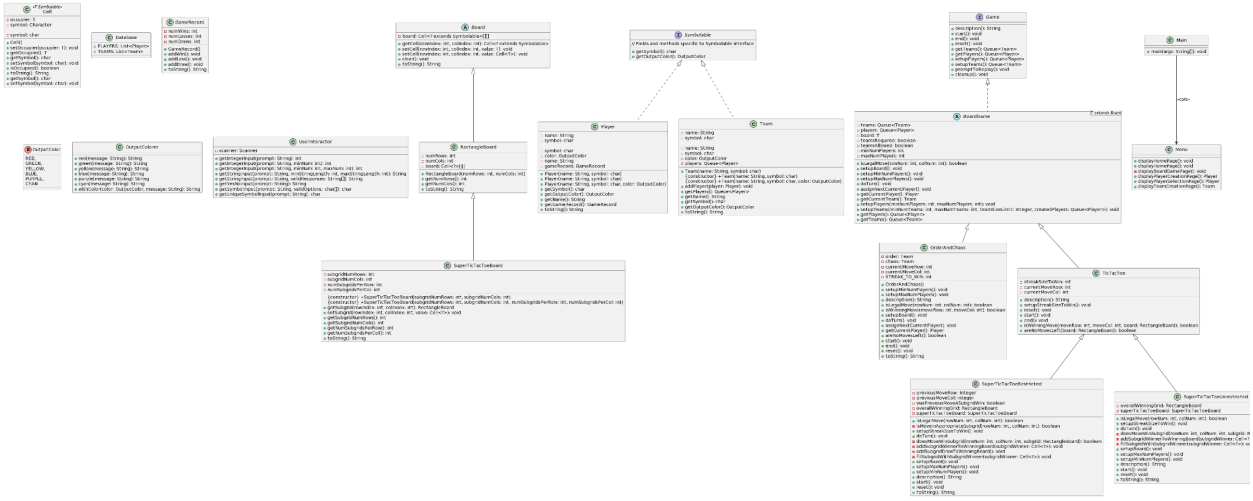
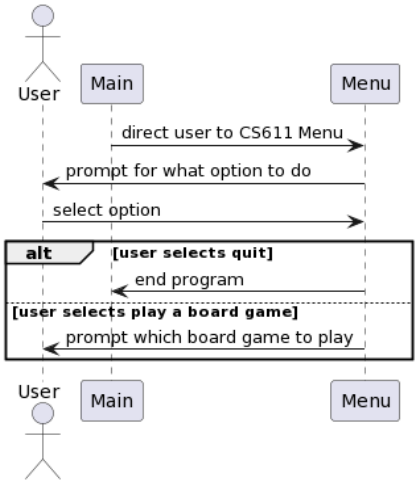
CS611 Assignment 2 Design Document

This document covers the high-level and compactly written object oriented design of my, Reshab Chhabra, CS611 Assignment 2. *Note: I am aware more classes can be made to abstract more content, but as I apply bias for action, I 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 "`Assignment2UMLDiagram.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.

Overall Organization Pattern

Originally, Board.java was a class that contained simple a rectangle-based board. But now, since games like Catan aren't rectangle in shape, I have made this class an abstract class and have made the previous Board.java from Assignment 1 into the class RectangleBoard.java. For Super TicTacToe Board, I extended it from the Rectangle class to make it called SuperTicTacToeBoard.java. This board includes subgrids of RectangleBoard, so if this class is used for other board games with rectangular boards, this can get renamed to MultiRectangleBoard.java in the future.

Game.java is an interface, and BoardGame.java is an abstract class that implements it. Then, TicTacToe and OrderOfChaos extend BoardGame, and Super TicTacToe (both variations I made) implement TicTacToe. What is important to note is that for the BoardGame, a type of board is passed in into each class that extends the BoardGame. For example, TicTacToe would be something such as BoardGame<RectangleBoard>. This makes it easier since each BoardGame has a generic Board field, which we can see in the provided UML diagram.

Another important thing to highlight is that Cell is a generic type that implements Symbolable – essentially, anything that can be represented as a single letter symbol on the Board is Symbolable, and Team and Player, as of now, are the classes that implement this.

The rest of the organization is indicated in the UML class diagram.