CS 520
Homework 1
Code Review, Architecture, & Design

Due: **Tuesday, September 29, 2020, 9:00 AM EDT** via Moodle. You may work with others on this assignment but each student must submit their own write up (with their name at the top), clearly specifying the collaborators (also at the top). The write ups should be individual, not created jointly, and written in the student's own words. Late assignments will not be accepted without **prior** permission.

Overview and goal

The goal of this assignment is to code review, redesign, and reimplement a Three in a Row game (conceptually Tic-Tac-Toe but with the rules of gravity applied[1]), according to the model-view-controller (MVC) architecture pattern.

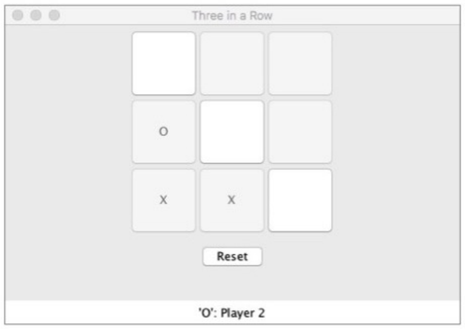

Figure 1: Screenshot of the 'Three in a Row' UI

Here are the basic rules of the Three in a Row game:

- Initially, the game board has each game block empty. The legal moves are in the bottom row.
- There are two players. Player 1 marks their blocks with 'X' while player 2 marks their blocks with '0'. Player 1 gets to make the first move.
- A legal move is to either an empty block in the bottom row or an empty block in an upper row on top of a filled block in the row immediately below.
- A player wins if they connect 3 of their marks (either 'X' or 'O') in a horizontal, vertical, or diagonal line. If neither player wins and all blocks are filled in, the game ends in a draw (or tie).
- After either player resets the game, the game goes back to its initial configuration.

The following repository provides a basic implementation of the Three in a Row game:
https://github.com/LASER-UMASS/cs520-Spring2020
This quick-and-dirty implementation satisfies some best practices but violates other best practices. It needs a major architecture and design overhaul. In contrast to the current version, your implementation should

[1]It is the 3X3 version of this game.

support possible extensions aiming to satisfy the open/closed principle (or at least improve encapsulation). Additionally, your implementation should enable individual components to be tested in isolation.

How to get started

1. Clone the repository https://github.com/LASER-UMASS/cs520-Spring2020 containing the *threeinarow* folder
2. Read the provided *README* in the *threeinarow* folder.
3. Use the commands to document, compile, test, and run the application from that folder.
4. Familiarize yourself with the original application source code contained in the *src* folder: src/ThreeInARowGame.java and src/ThreeInARowBlock.java.

Code review

You are expected to code review the original version of the application focusing on the expected behavior (e.g., encapsulation, extensibility, testability) instead of coding style (e.g., amount of whitespace, if vs switch statement). In particular, you need to identify 3 cases where best practices are satisfied. Additionally, you need to identify 3 cases where best practices are violated. Your code review should use the following pattern **for each identified violation**:

- Brief summary of this issue (a few keywords)
- Explanation of the issue (could refer to general principles or poor design choices with respect to the desired extensibility and testability)
- How to fix it (a few sentences)

In the second homework, the 3 identified violations WILL need to be implemented.

Architecture and design

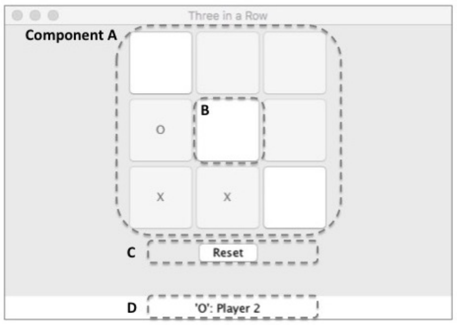

Figure 2: Main components of the ‘Three in a Row‘ UI

MVC architectural pattern Identify the following in the Three in a Row UI:

- Component A: View, Controller, or both?
- Component B: View, Controller, or both?

- Component C: View, Controller, or both?
- Component D: View, Controller, or both?

Identify the original application source code (such as classes, fields, and methods) corresponding to the:

- Model
- One view (from above)
- One controller (from above)

Observer design pattern For the original application, the *Observer* design pattern is being applied for the relationship between a View and its Controller. (There are two such pairs.) Identify one java class that corresponds to an *Observable*. For that *Observable* class, identify the java class that corresponds to its *Observer*. Identify the implementation of the *update* method for that *Observer*.

Deliverables

Your submission, via Moodle, must be a single document (plain text or PDF) named *hw1.txt or hw1.pdf*, containing:

1. Your full name and any collaborators (both at the top)
2. Code review comments
3. Identification of the *MVC architecture pattern* in the UI and the application code
4. Identification of the *Observer design pattern*