Final Project Report

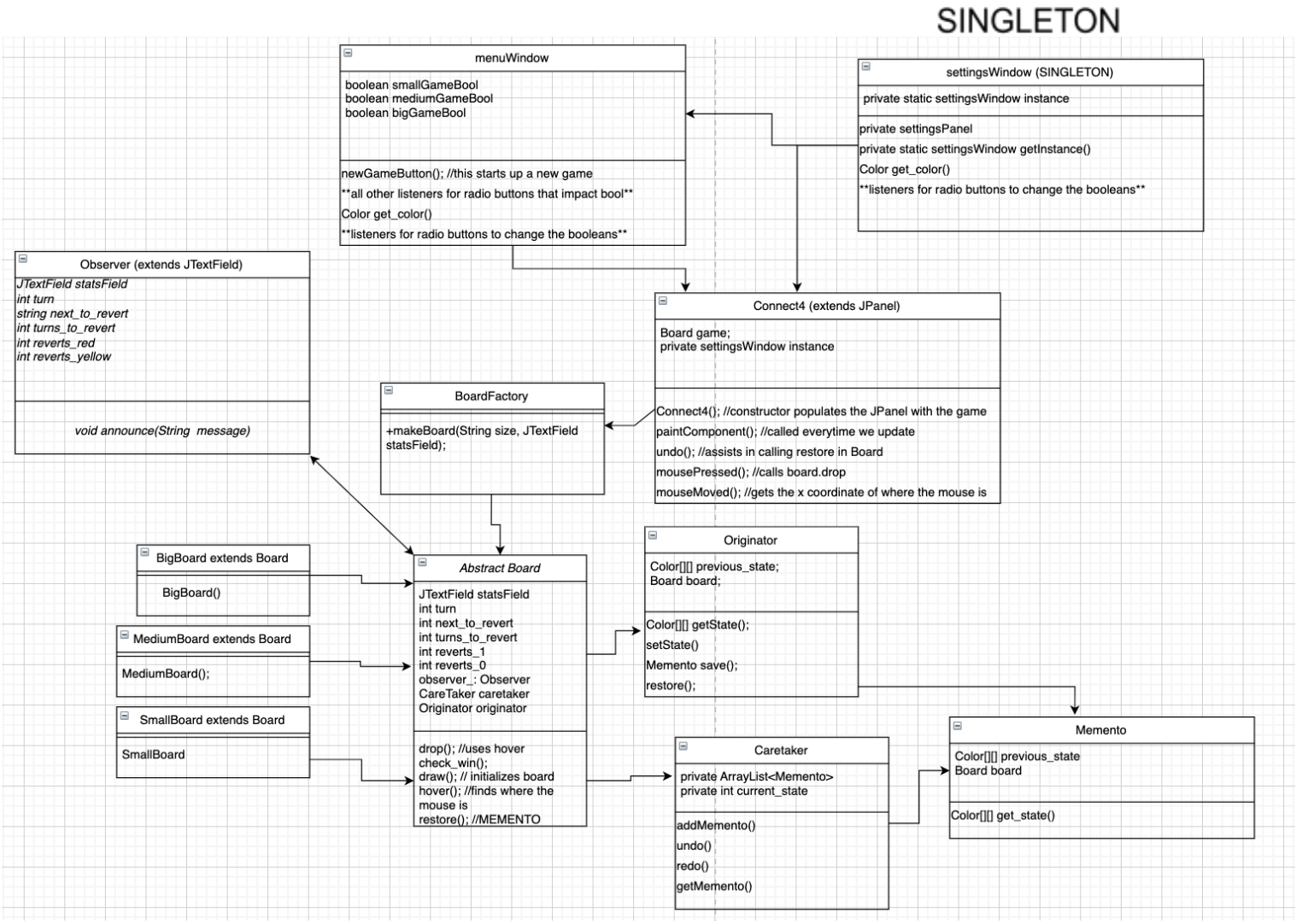
Names: Cori Carver, Conlan McConvey and Glenn Holzhauer
Project: Connect-4

Final State of System Statement

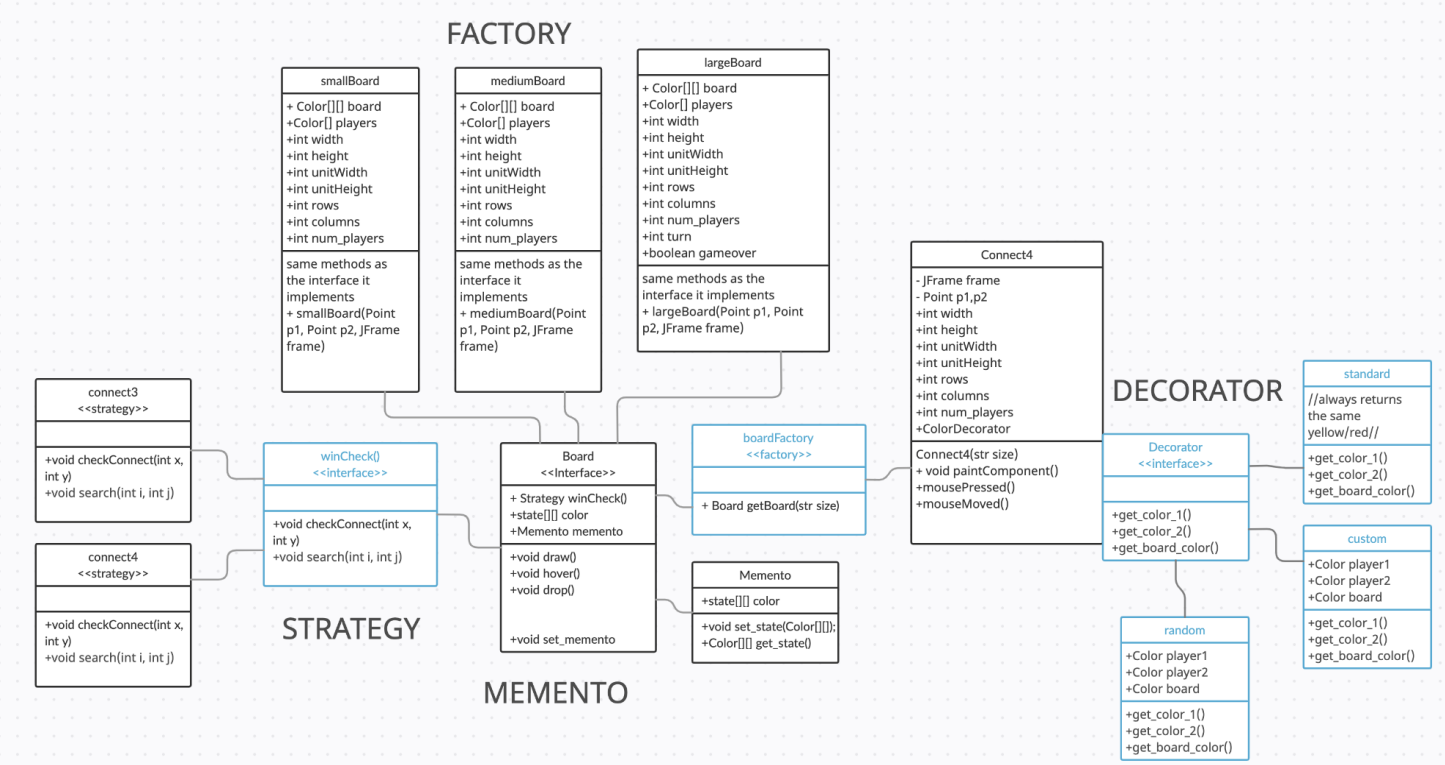
Our project had several adjustments over the course of our development cycle. Initially, we had thought we would implement our Connect 4 game on a web platform and use persistence in the form of a database. We realized in our process that maintaining our code on a web platform and including a database would add complexity and decided it seemed more appropriate for the purposes of this class to focus on the patterns we were implementing, so we built it to run as a local application. After building out our classes, we also realized a decorator pattern and factory pattern would be challenging to have alongside each other because of our selected implementation for the factory pattern, so we decided the factory pattern would be more meaningful to our game. We also needed to eliminate our strategy pattern because it no longer fit our class implementations and felt forced. Otherwise, most of the functionality we felt was most valuable and interesting for the application was implemented, including a memento “undo” function.

Final Class Diagram and Comparison Statement

Final UML Diagram:



Initial UML Diagram:



Key Changes: Initially, our game was going to be very dependent on a database and the ability to save a game in progress and return to it later. As our game evolved, we decided to place less emphasis on the database functionality, and instead focus more on implementing the patterns. This change can be seen in the drastic differences between our initial and final UML diagrams. We also decided to forgo the login and user account functionality in favor of a more developed and appealing UI.

Third-party code vs Original code

- https://www.codejava.net/java-se/swing/jpanel-basic-tutorial-and-examples - I used this resource to get a better understanding on how to use JPanel
- https://www.youtube.com/watch?v=E6TPk2GVuIY&t=361s – helped get an understanding for working with a Swing Application and provided the groundwork for our win condition
- https://www.youtube.com/watch?v=8hvvyJPNaBE&t=401s – helped get an understanding of how to use the Memento Pattern

Statement on OOAD Process for Project

Our team learned a lot about how to design, create and facilitate object interactions in this project. First, we experienced iteration-heavy design throughout this project,

constantly having to adjust our approaches to our patterns when we realized our design did not factor into consideration some object's relationship with another object. This meant lots of communication to ensure that we were not breaking each other's contributions, and sometimes spending more time on getting our code to all be compatible. Second, we also experienced the consequences of failures to encapsulate code well. When our code was not well encapsulated, we saw that version control and maintenance with other developers became more challenging, and as we developed our code base the time we had to take to correct these issues became more extreme. Third, we had issues with our strategy pattern implementation because we initially thought the structure would be simple, but found as we created our project that it was not applicable for the purpose of detecting 3, 4, and 5 consecutive pieces. In the future, we will need to carefully consider how patterns might conflict with one another as we design our software.