Onitama

Team Onitama: Alex Keyser,
Caleb Lefcort, Colin McClelland

Project Overview

- We aimed to create a fun, easy to understand, and replayable re-creation of our game that anyone can enjoy
  - We chose a lesser-known game for our project, Onitama
  - This gave us a fresh look into rules that we were not already familiar with, as well as let us present a game that deserves more recognition
- The main features we focused on, were a accurate representation of Onitama, an intuitive interface, and a character/icon select system!
  - There were some limits, as we could only make so many different sprites for the project in this time
  - We don't have a tutorial, but we provide the rules as an option to look at in case you don't know how to play

Game Description

- Onitama is a turn-based two-player game that involves a good amount of strategy to overcome your opponent.
- The goal of the game is to take your opponent's master pawn, or reach the center of their side of the board
- Movement revolves around 5 random cards that have available moves indicated on them.

Project Requirements

- Functional
  - Cards and Pieces are separate, but work together
  - Turn rotation and access to cards along with it
  - Win conditions

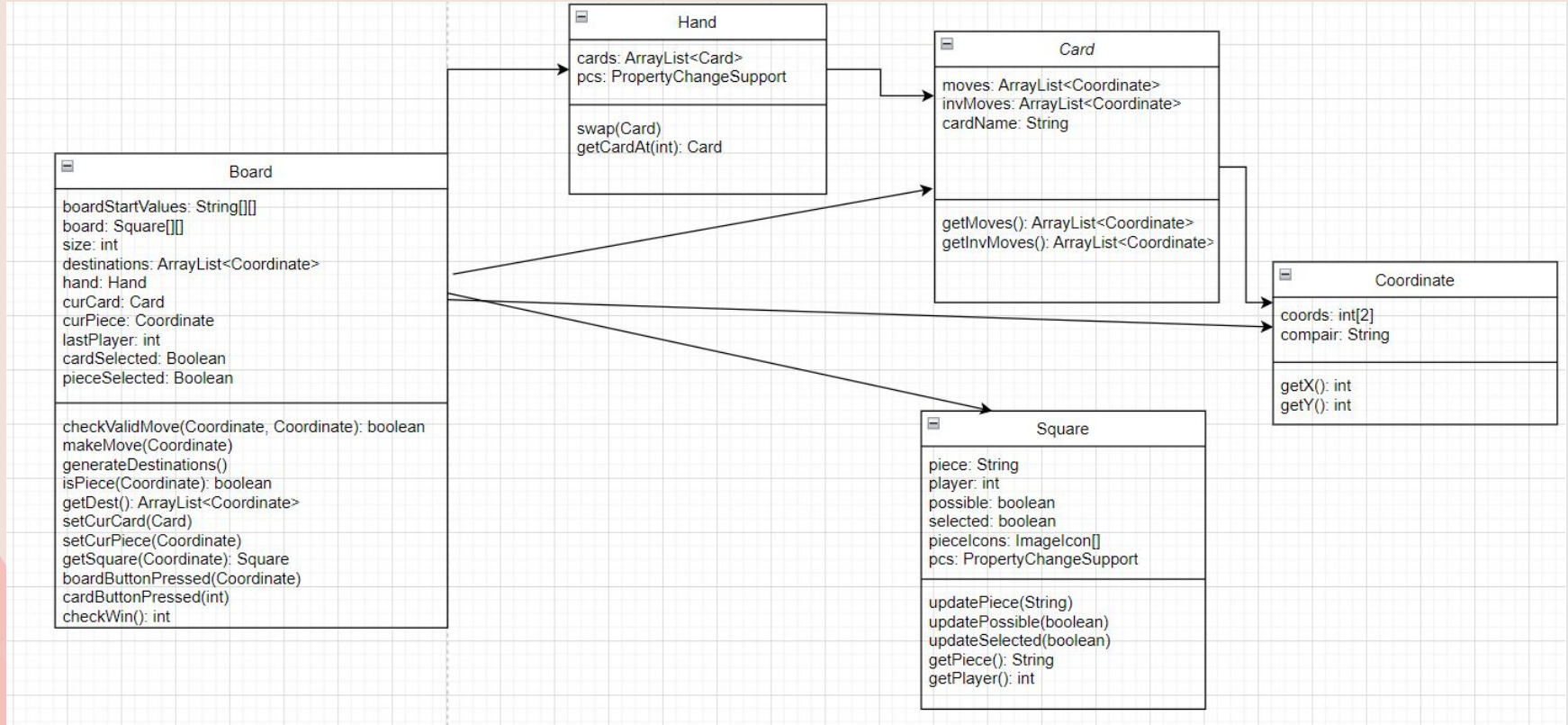
- Non-Functional
  - Character/Icon Selection
  - Cards move around the board
  - Moving Icons
  - An easy-to-understand display

Project Solution Approach

- What are the major components in your solution design?
  - The Board is a 2d array of Squares and each button in the grid of buttons observes its respective square for changes.
  - The game logic is mostly run through the board class and the buttons alert the board that they have been selected and their coordinates.
  - The hand is an arraylist of cards and each card button observes its respective card for changes.

- What game/ui features did you really identify and work towards in your take on your game?
  - A lot of effort was put into piece/ card selection.
  - Attractive presentation of the game board.
  - Color selection for the players.

UML Design here - How did you structure your solution?



Team Collaboration Approaches

- Let us know how your team collaborated to make this happen
  - We used Imessage to communicate and Google Docs to share non coding work
  - We did not use the GitHub issue page very much but instead relied on verbal/ written agreements.
  - Git branches were very useful for working together and we quickly learned how to resolve conflicts.
  - Try to design the code in a way that each member can work on their own tasks separately.
  - We did the majority of our work alone but would look over code together if we had a tricky bug or needed to make a major change.

Testing, Validation, and Acceptance Plan

- Testing Approaches
  - We relied heavily on ad hoc testing
  - Time constraints
  - User testing for the finished product

- Is it deliverable?
  - We kept the requirements in mind as we built the game
  - Reviewing the finished product
  - User Acceptance Testing
  - Personal standards

Live Demo

- Let’s play!

Summary

- Future Work
  - More customization options
  - Expand upon the existing features
  - Downloadable app?

- Closing Thoughts
  - Pleased with the finished product
  - Lifespan of the project
  - Working as a team
  - Going Forward