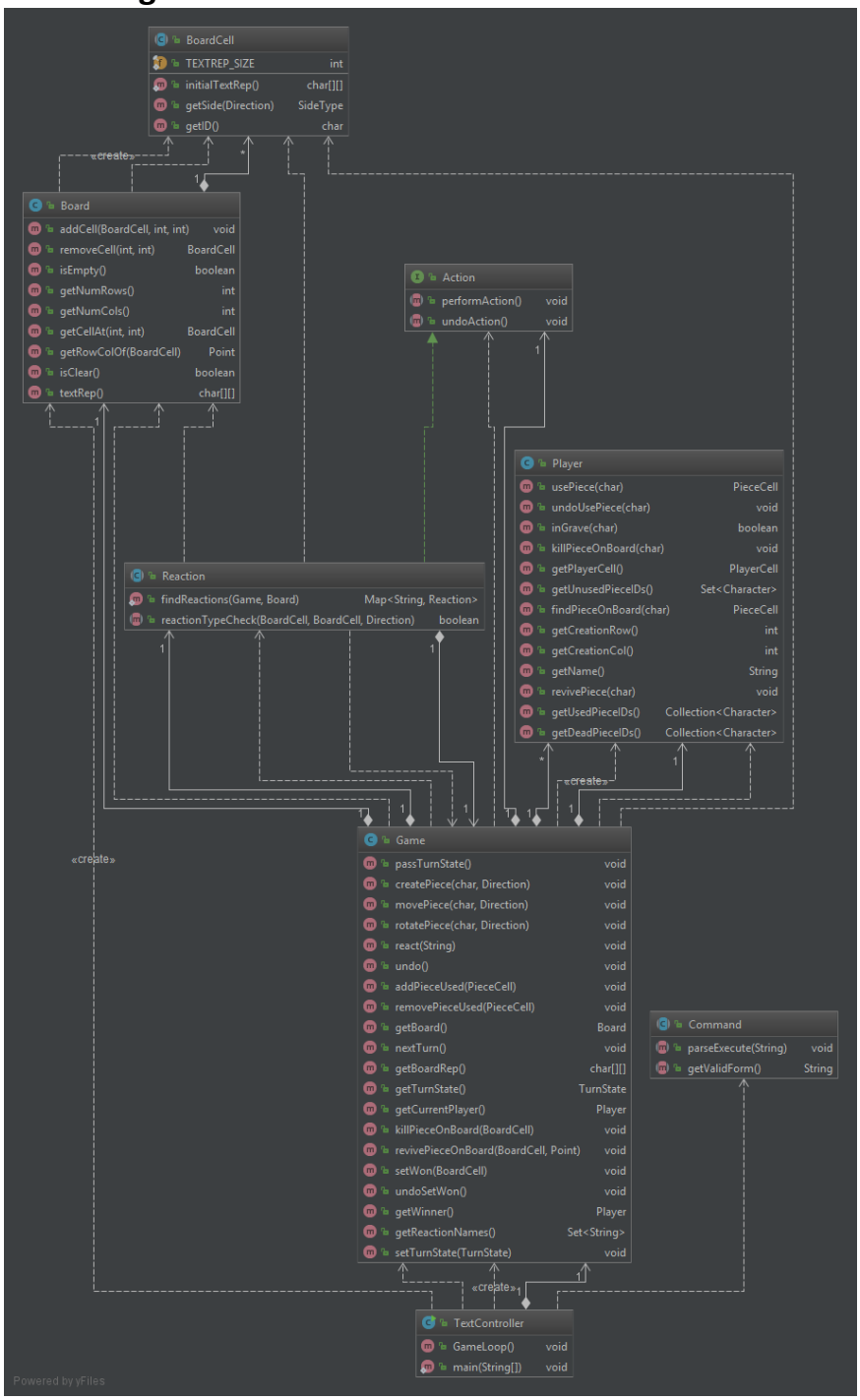
Bryn Bennett
300343196

SWEN222 Assignment 1

Note this document contains all the requested sections

Class Diagram



Class diagram with the different kinds of exception, action reaction and minor classes removed

Bryn Bennett
300343196

Design Report

This app loosely follows MVC (Model-View-Controller), I say loosely as it is split into a model package the manages the game rules, states and behaviour, with Game being the class that brings all this functionality together. The controller is in the controller package under TextController. This takes the user input processes it and calls the relevant methods on the model to update the game.
The app starts to be looser with MVC when it comes to the view. For this app, the view is a plain text interface. Rather than make this its own independent module it is a bit more spread out with each object that should be visible in the view being responsible for creating its own textual representation as a 2d array of characters "textRep()". I used this approach for the view as it seemed to suit the nested object structure of having a board containing pieces that then have sides and different items on those sides.

The Game uses a command pattern to perform actions, with CreatePiece, MovePiece RotatePiece and the different reactions being various kinds of action that can be performed. This suits the requirement of an undo as each action is stored on a Stack to allow for undoing. Actions have an undo method to 'undo' the events that occurred when initially performing the action.

Good Code Highlight

Using abstract methods with enums such as the TurnState and SideTypes. Having these abstract methods means that each enumeration must provide functionality that can be unique to it. This is beneficial when it comes expanding the app as there aren't as many if statements or switches that check for a enumeration and act on that, and so if the expansion required new enumerations these switches or ifs shouldn't be present and so needing to be updated.

On a small scale take SideType where each enumeration has its own textRep() that returns the character that will represent it based on a given side. If different types where added such as Bows etc, then it will have its own textRep() added when it is coded rather than tracing down and updating if statements or switches.

On a larger scale TurnState uses it in conjuction with an interface to map different states (enumerations) to different commands. This means that adding new states and updating the internal interface makes the IDE ask the coder to also update the CommandMap. I think this will proof useful with the graphical user interface as the CommandMap can be swapped out.

Testing Discussion

While creating the initial functionality and structure for the app, I had not started making tests. This was because I had not fully settled or decided if this was an implementation for the assignment that would work. However once, I had the base structure of the app, I began creating tests to ensure that objects such as the player behaved as expected and managed the pieces they had.
When it came to coding the different actions and reactions my development style moved more to test driven development. For each action, I created a series of tests that would pass if the actions basic functionality worked. When written these were failing tests, but using them allowed me to gauge the suitability of my action implementations and where and if things were going wrong.

The more complex tests such as those for actions and reactions are done through the text controller, this allowed me to easily display the games board and diagnose at which step the tests where performing an action that was not behaving as expected. This combined with the debugging tool made finding where issues were much faster and easier.

Bryn Bennett
300343196

I also frequently tested the app manually through the text interface, this helped me understand the reactions that where occurring and picked up on issues I hadn’t thought to test. For these issues, I then wrote failing tests to help in fixing them.