Project 7

Project Members: Austin Cha
Rees Labree
Jonathan Wick

Project Name: PyGo - The Game of Go

Github: https://github.com/reeslabree/pygo

Project Video:
https://drive.google.com/file/d/1tnqHZeHTL1hmLKoB6T4G_4VxS04as-g6/view?usp=sharing

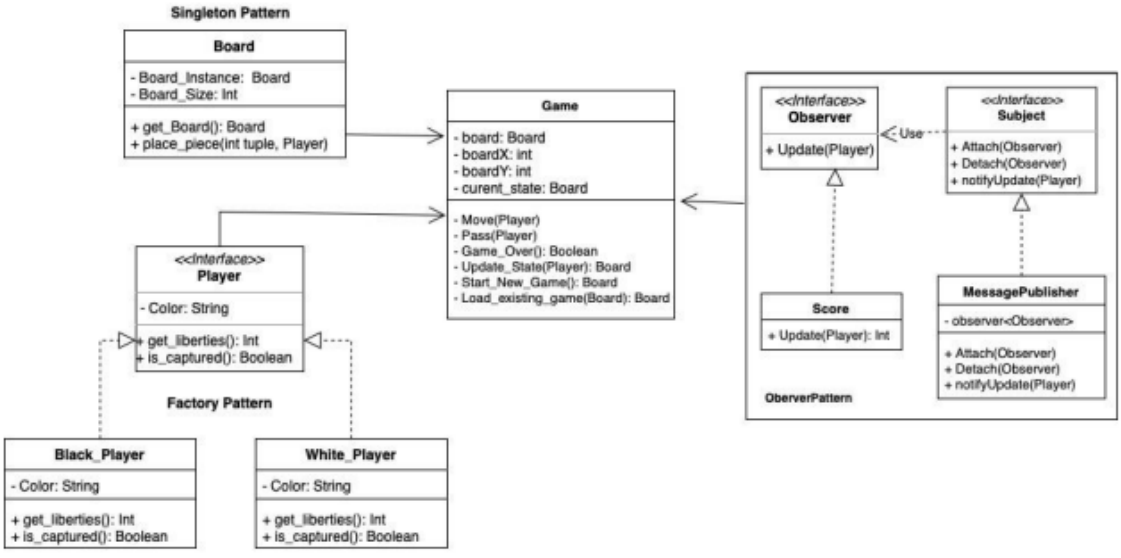
---

Final State of System

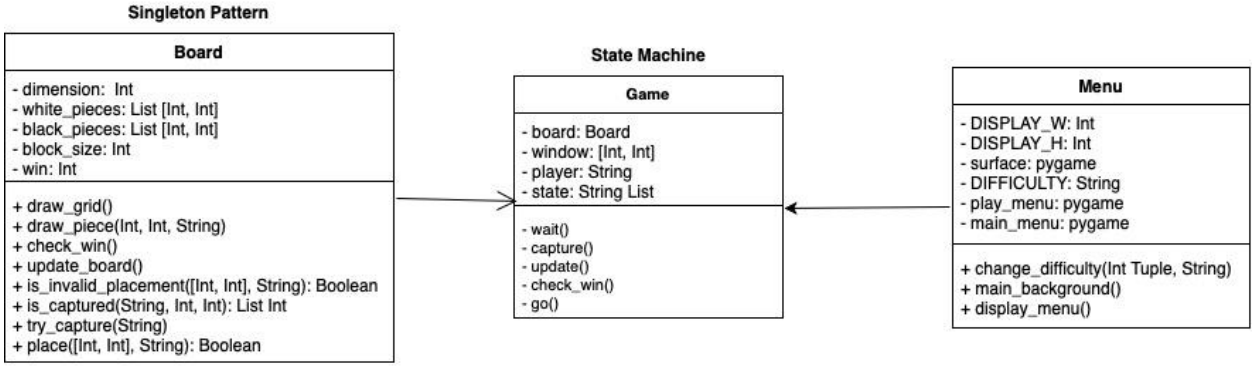
We have implemented all planned requirements from project 5, however the graphical design of the game itself isn't as clean as we would like it to be, and we did not implement players who could be named. We focused a lot on the technical aspects going on behind the scenes. Instead of tracking scores with the Player objects, we instead tracked score inside the Score object which implements a strategy to determine if prisoners are captured or not, and territory captured. The Menu and Game follow State patterns, with the menu using the concept of a Game Loop pattern to pass between the different menus, and the Game and board using a Singleton pattern. Score.py is implemented as a Strategy pattern.

UML Class Diagrams

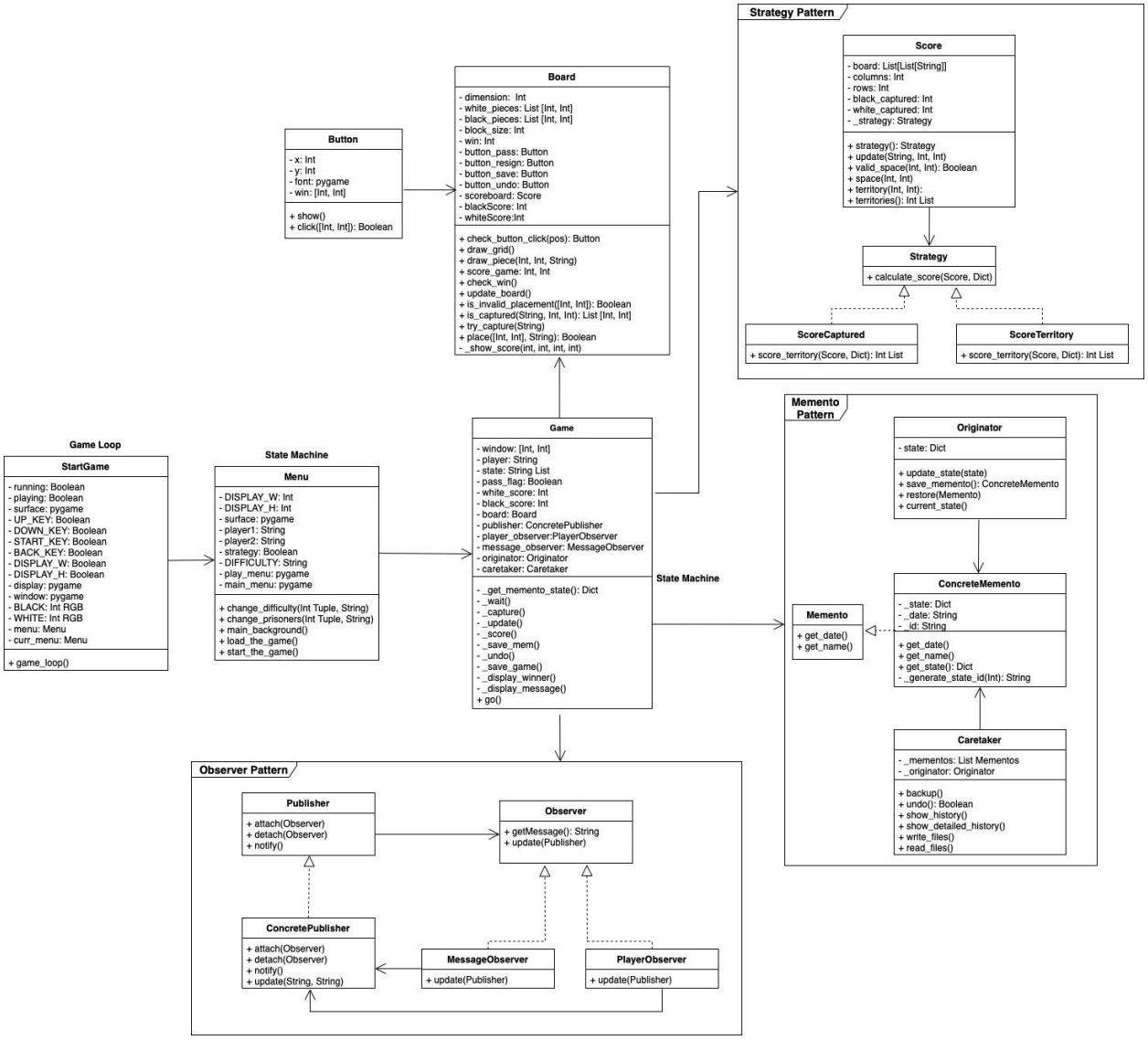
Project 5



Project 6



Project 7



UML class diagram - Key Changes

There are a few changes and design patterns we added from our initial plan in project 5. First, we used a memento pattern to save and load a pre-existing game. Originally, we were going to have a function in our Game object that saved the player pieces of the existing game. However, with the memento pattern, we were able to restore the entire Game object to the previous state. With the game of go, there are many ways to calculate the score. Therefore, we also implemented a strategy pattern to calculate the different ways players could get points. This way we can dynamically assign which scoring algorithm to use.
Finally, we did not use the factory patterns for the players as both players had the same behaviors.

Third-Party code vs Original code Statement
Additionally documented in the codebase

- **Observer.py**
  - Used the template from refactoring guru
  - https://refactoring.guru/design-patterns/observer/python/example
- **Memento.py**
  - Used the template from refactoring guru
  - https://refactoring.guru/design-patterns/memento/python/example
- **Score.py**
  - Used an algorithm to solve territory captured in the game of Go taken from Source: https://exercism.org/tracks/python/exercises/go-counting/solutions/CFred
  - This was adapted into our strategy pattern and score keeping objects. Source: https://www.tutorialspoint.com/python_design_patterns/python_design_patterns_strategy.htm
- **Menu.py**
  - Used pygame_menu documentation example for menu reference. Source: https://github.com/ppizarror/pygame-menu/blob/master/pygame_menu/examples/game_selector.py

---

OOAD process

- Our group could have benefited from more communication, as we often had redundancies in the code that needed to be taken out towards the end of the project.
- We could have benefitted from more testing, as we discovered a flaw in our scoring system late into the project which required us to put significant work into resolving.
- We supported and mentored each other when we got stuck to implement effective solutions and patterns.