OOP - Assignment 5

Circus of Plates

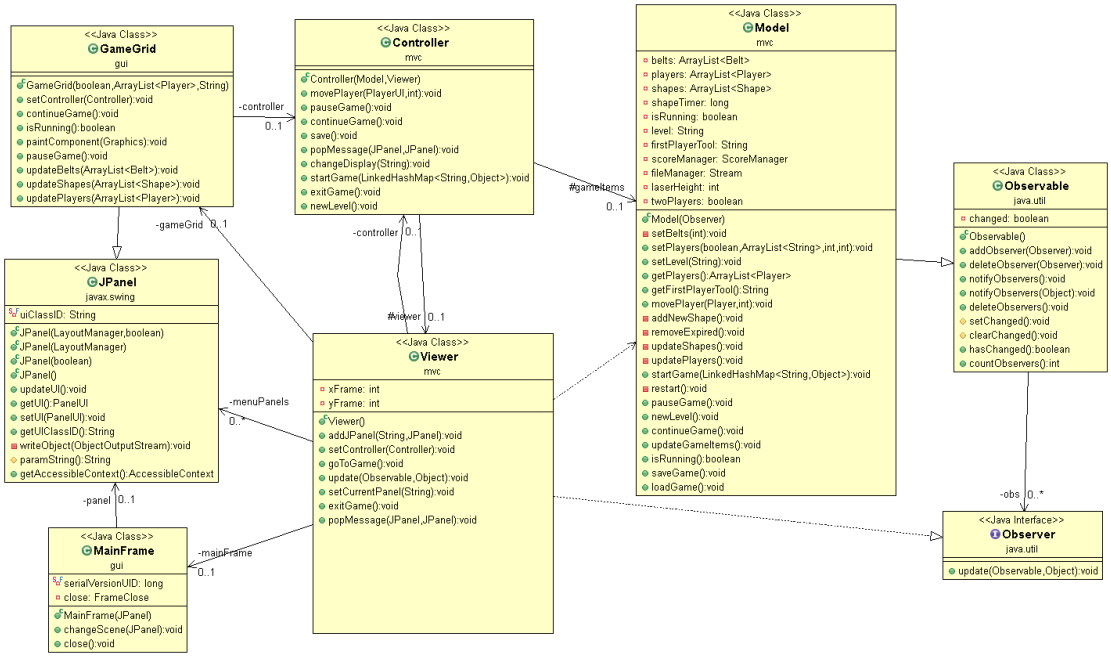
Abeer Ahmad - Shrouk Ashraf - Moustafa Mahmoud

Used Design Patterns:

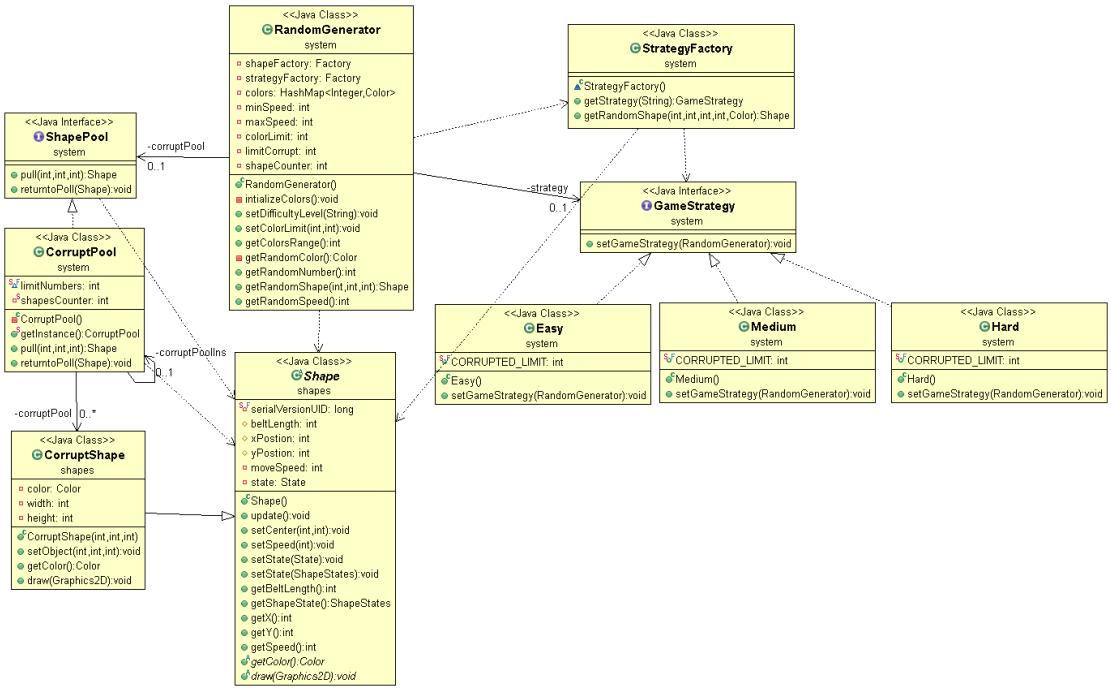
1. State Design Pattern:
    - used to store the current state of each shape, could be one of the following 4 states: (OnBelt, Falling, Captured, OnGround).
    - allows each shape to have a unique indicator to its state, that is updated automatically.
    - the state transition process is totally isolated from the ***Shape*** class, all a shape needs to know is its current state, without going deep in details about how it was obtained.
2. Strategy Design Pattern:
    - used to determine which strategy the game will follow, in other words, which level of difficulty the game will sustain.
3. Factory Design Pattern:
    - used to return an instance of a specific class among different classes, according to a given key (condition) to determine of which class that instance will be.
    - this factorization is done separately from the class that demanded an instance.
4. Object Pool Design Pattern:
    - used to make use of previously created costly objects, without the need to create new instances when the number of created instances meets the max limit.
5. Singleton Design Pattern:
    - used to create one and only one instance of some classes, to guarantee that no other instance is interfering with its job.
    - in our case, we need both ***ScoreManager*** and ***CorruptPool*** to the only single instances in our game.
6. Observer Design Pattern:
    - used to make a specific class observe another class (keep an eye on it); so that whenever any change occurs in the observed class, the observer class is notified to respond.
    - in our case, we need the ***Viewer*** to observe the ***Model***, when the ***Model*** is changed, the ***Viewer*** is notified.
7. Dynamic Linkage Design Pattern:
    - used to load shapes dynamically on runtime.
8. MVC Design Pattern:
    - used to connect the ***Model***, ***Viewer*** and the ***Controller*** to each other.
    - the ***Controller*** receives updates from the ***Viewer*** and notifies the ***Model***.
    - the ***Model*** can notify the ***Viewer*** directly (using the **Observer** pattern).
9. Snapshot Design Pattern:
    -
10. Facade Design Pattern:
    - used to be in control of different related classes using one simple class that can reach them all.

UML Diagrams:

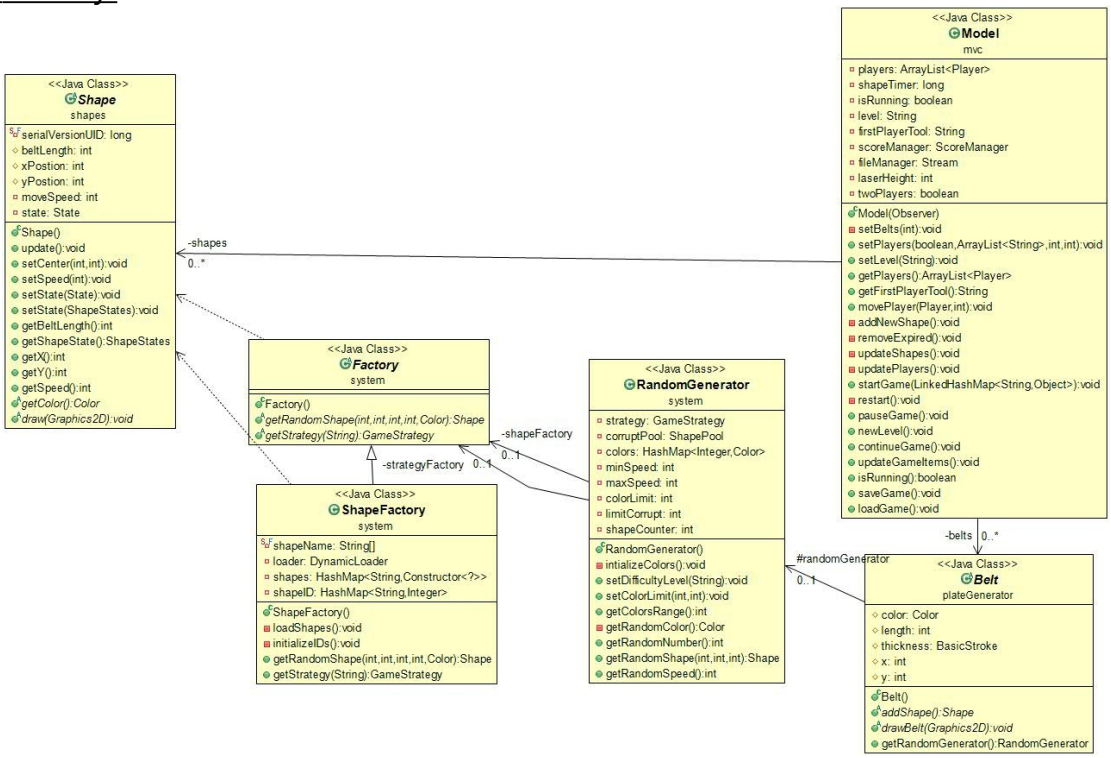
1. MVC - Observer - Facade:



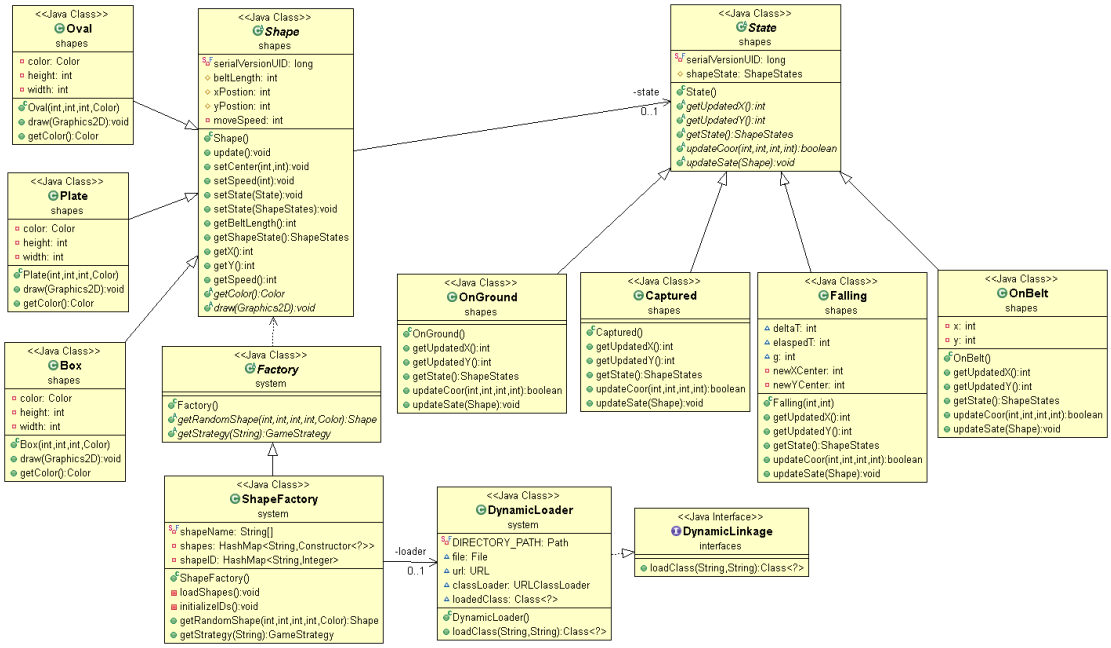
2. Strategy - Object Pool - Sigleton:



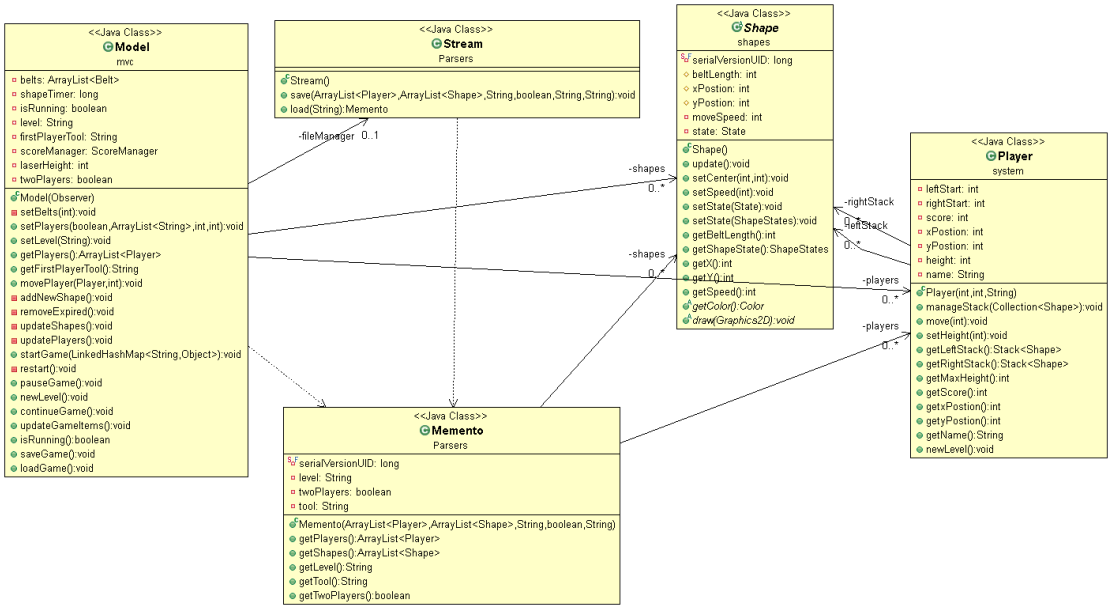
3. Factory:



4. State - Dynamic Linkage:



5. Memento:



6. Game Loop: