Faculty of engineering

CCe 2020

Version 0.5.2

May 21, 2017

Project 3 report :

Circus of plates Game..

Developed by :

Mohamed Ashraf Ramzy 3830

Mohamed Salama 3827

Ayman Marrouf 4331

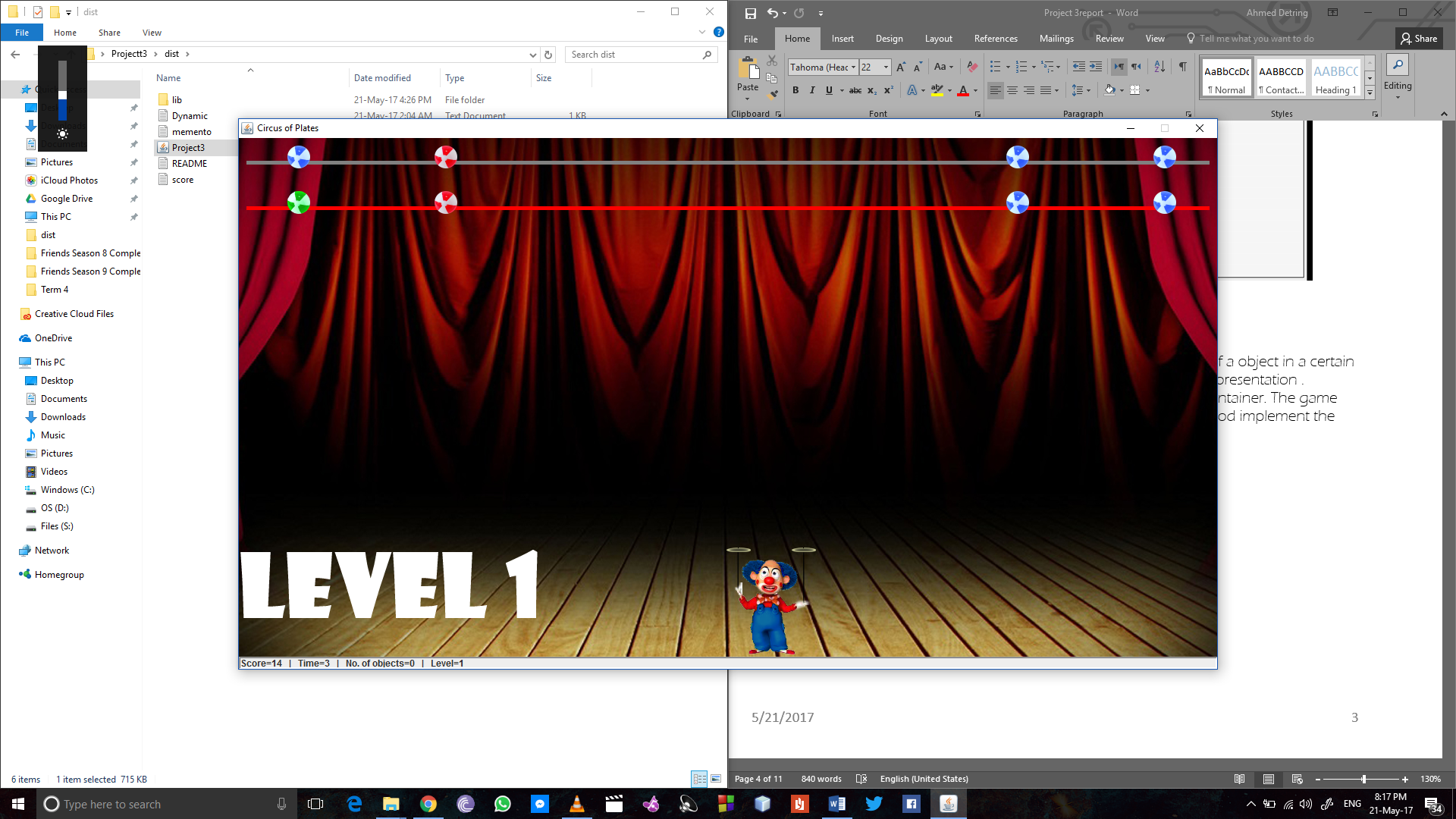
Ahmed Detring 3892

Reyad HusseinMelies 4184

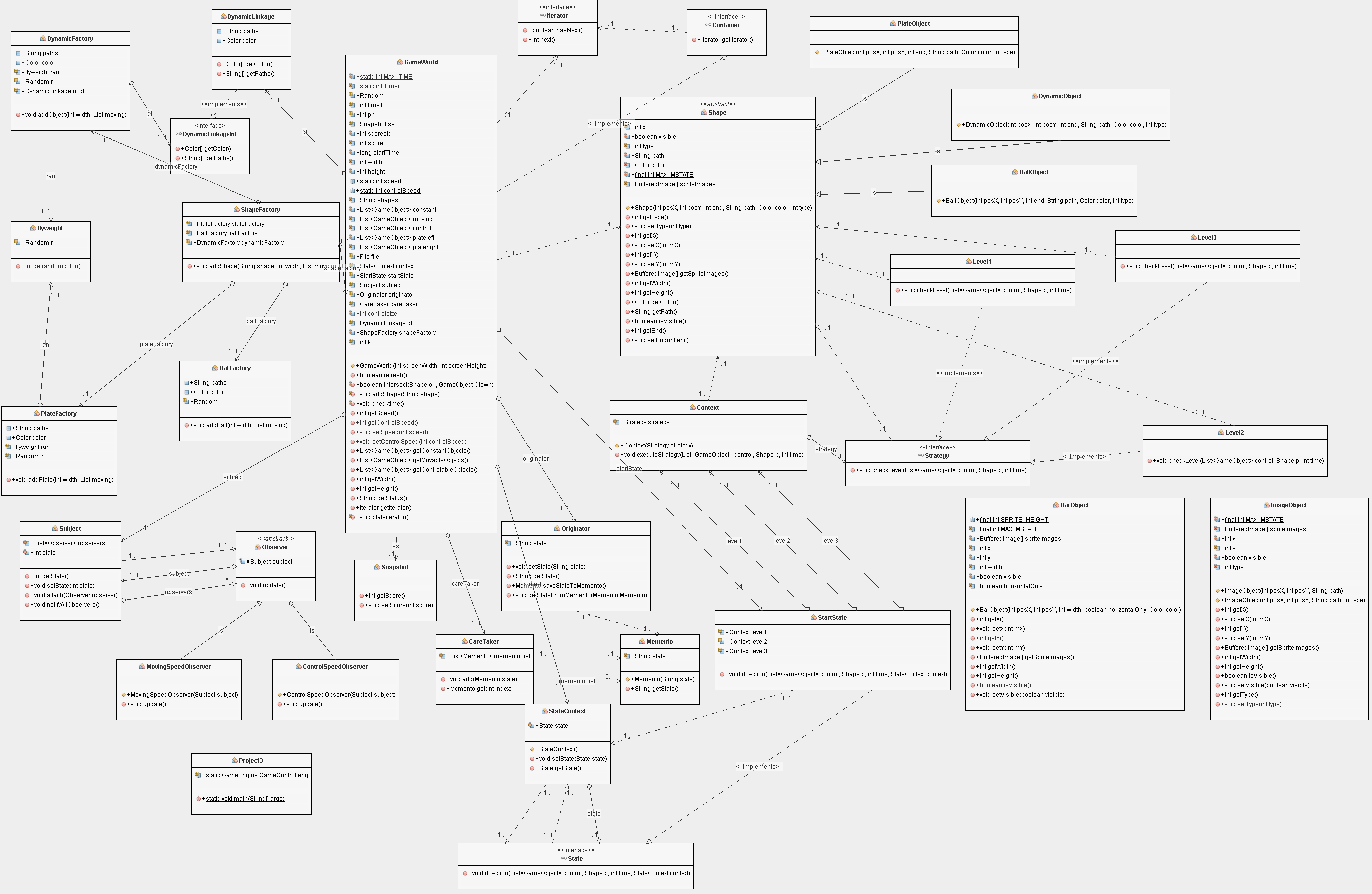
Amr Saleh 4023

**Description& Featues :**

The circus of plates is a single player game, developed for our final project. Where the clown catches, the falling plates form above in his hands, the plates are randomly colored, so if he catches 3 consecutive plates of the same color, his score increases by a point and the 3 plates disappear to make room for new ones.After each 30 second, the difficulty level increases gradually, adding more plates per second dropping to his hands. The game ends when the plates in one hand reach the top of the game window. The clown is moved by the right/left arrow keys .

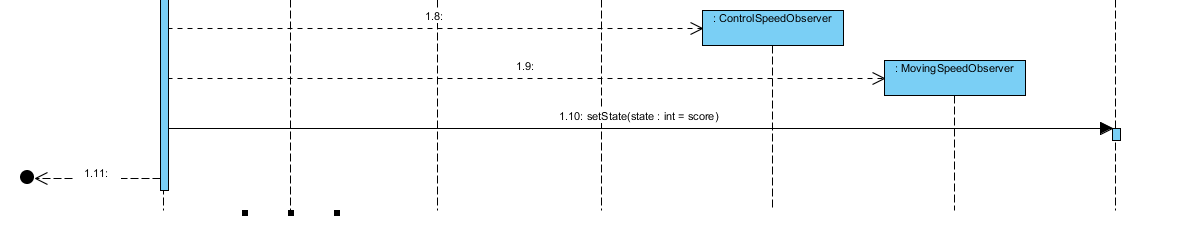


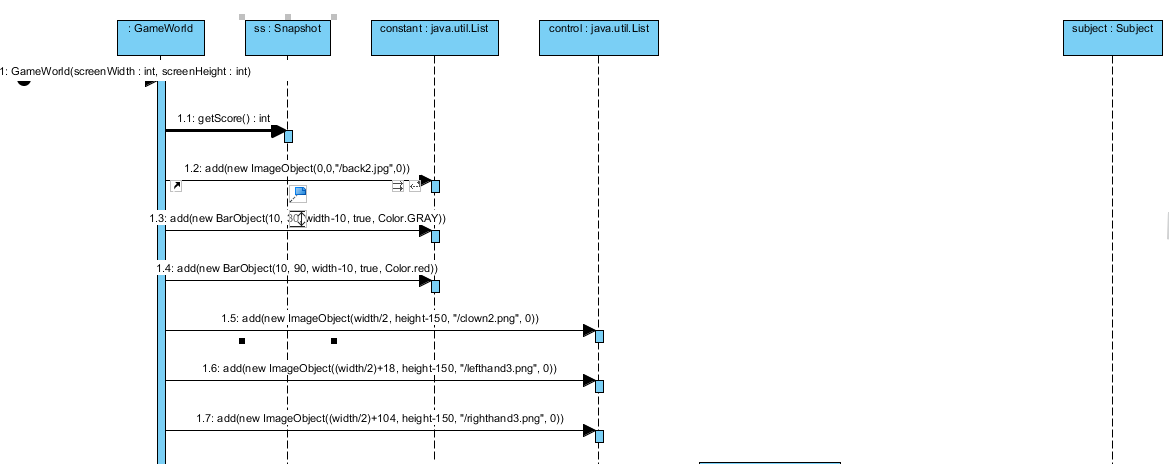
**Class Diagrams:**

The UML diagrams for classes and interfaces are as follows:

**Sequence Diagrams:**

The sequence diagrams for classes and interfaces are as follows

****



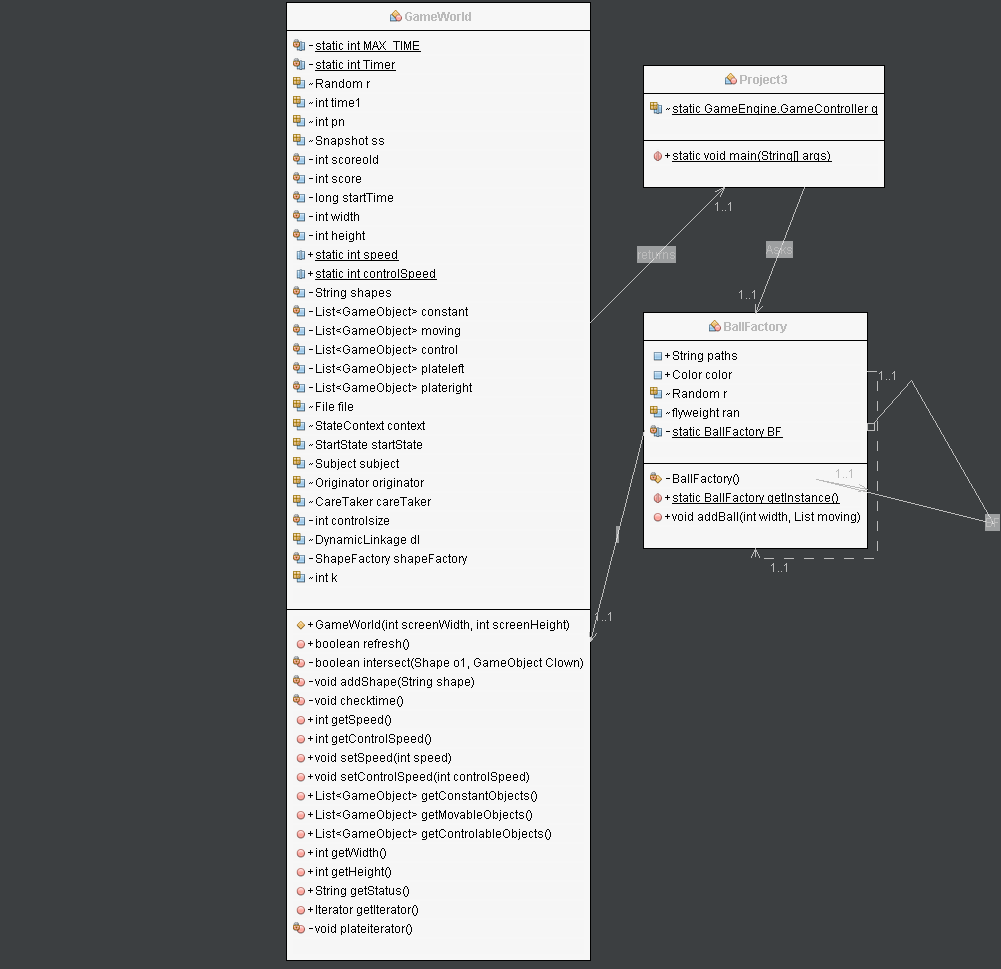
**The Design Patterns :**

The used desigin patterns are :

1. Singelton:

This pattern uses a single main class , responsible to create an object and and makes sure that only one object of it is created so it’s used only one time .

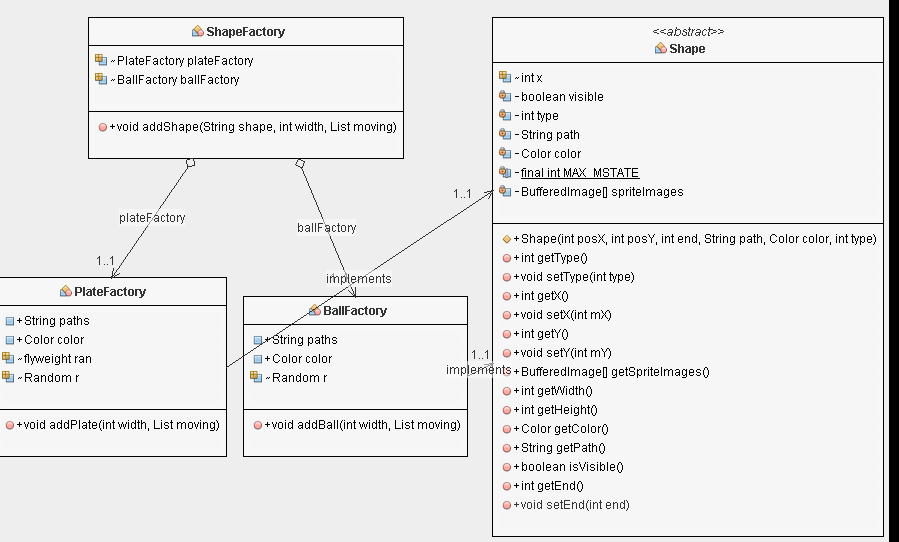
The singeltone class here is the ball factory and the single static object it calls is the game engine that takes and instance of the ball factory and sends it to game world, where it’s responsible to send yo the main method to start the game in a new game world of set resolution.



2-Factory :

The factory pattern uses factory methodes like ball factory and plate factory to create objects of them by calling these methodes which are then implemented in a base class.

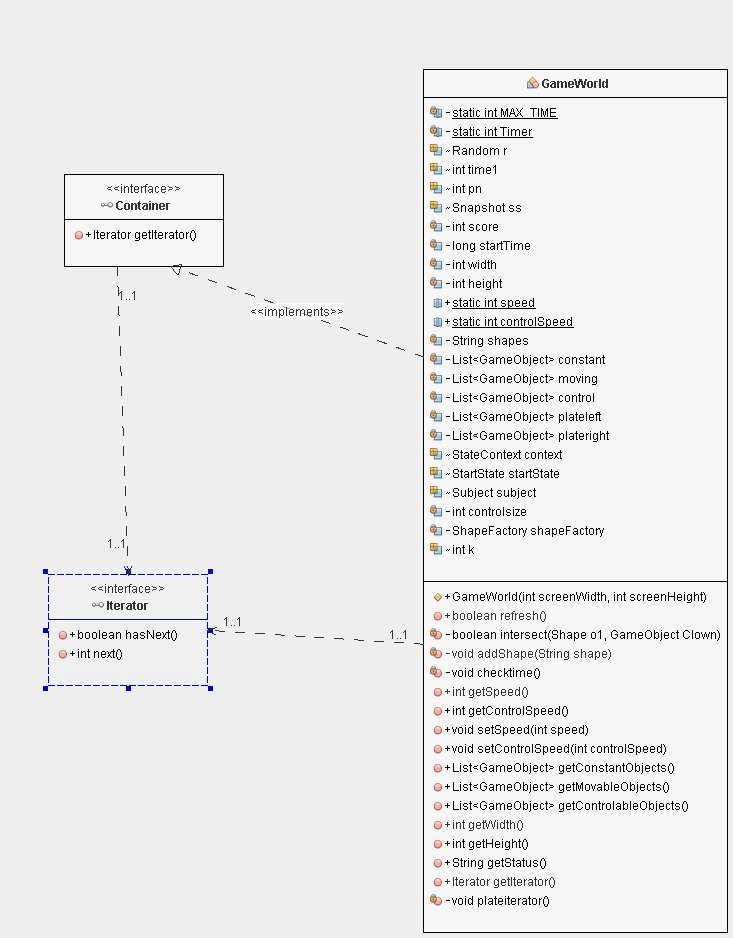
The plate and ball factory implements the shape class and the shape factory has an instance of both ball and plate factory.



3-Iterator :

The iterator pattern is used to get a way to access the elements of a object in a certain sequential manner without any need to know it’s underlaying representation .

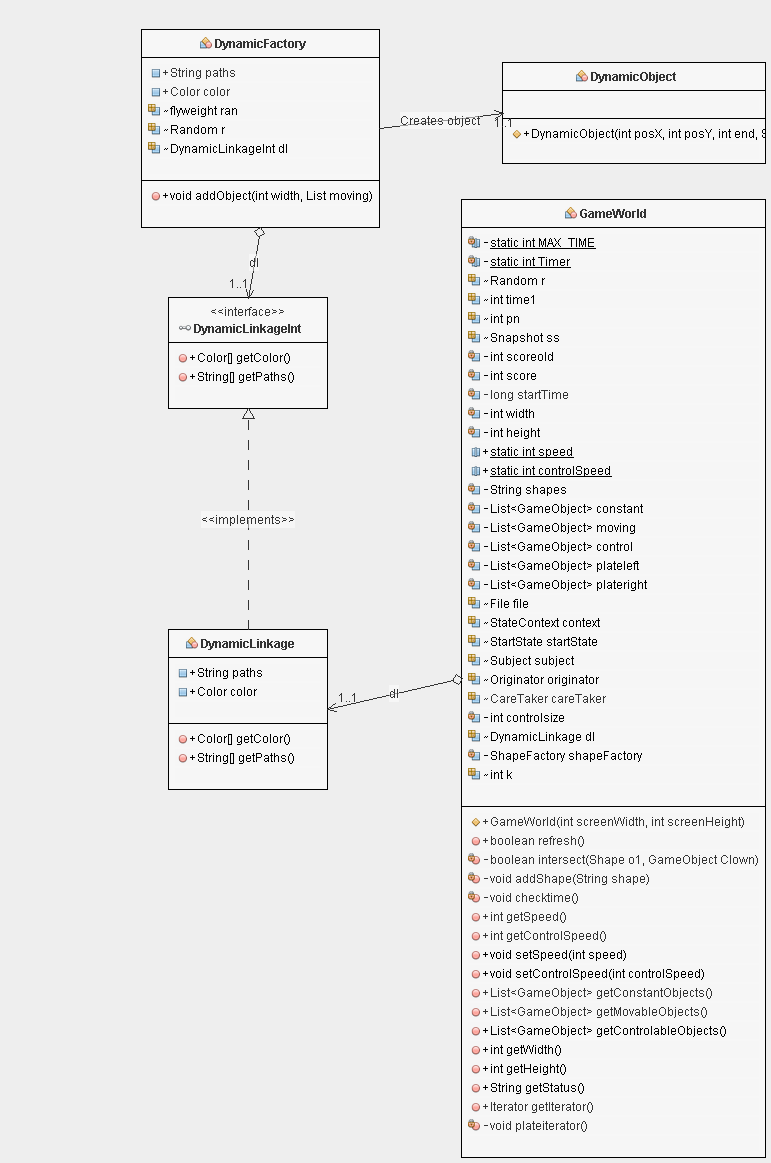
There are 2 interfaces in this pattern , the iterator itself and the container. The game world here implements the container and the plate iterator method implement the iterator.



4-Dynamic linkage:

The Dynamic linkage simply reduces compile-time restrictions on behalf of more run-time freedom regarding class implementation. The procedure allows change in implementation of classes at run-time, without neither re-compiling nor stopping the program. The dynamic linkage here was used to import new shapes rather than the normal ball and plate.

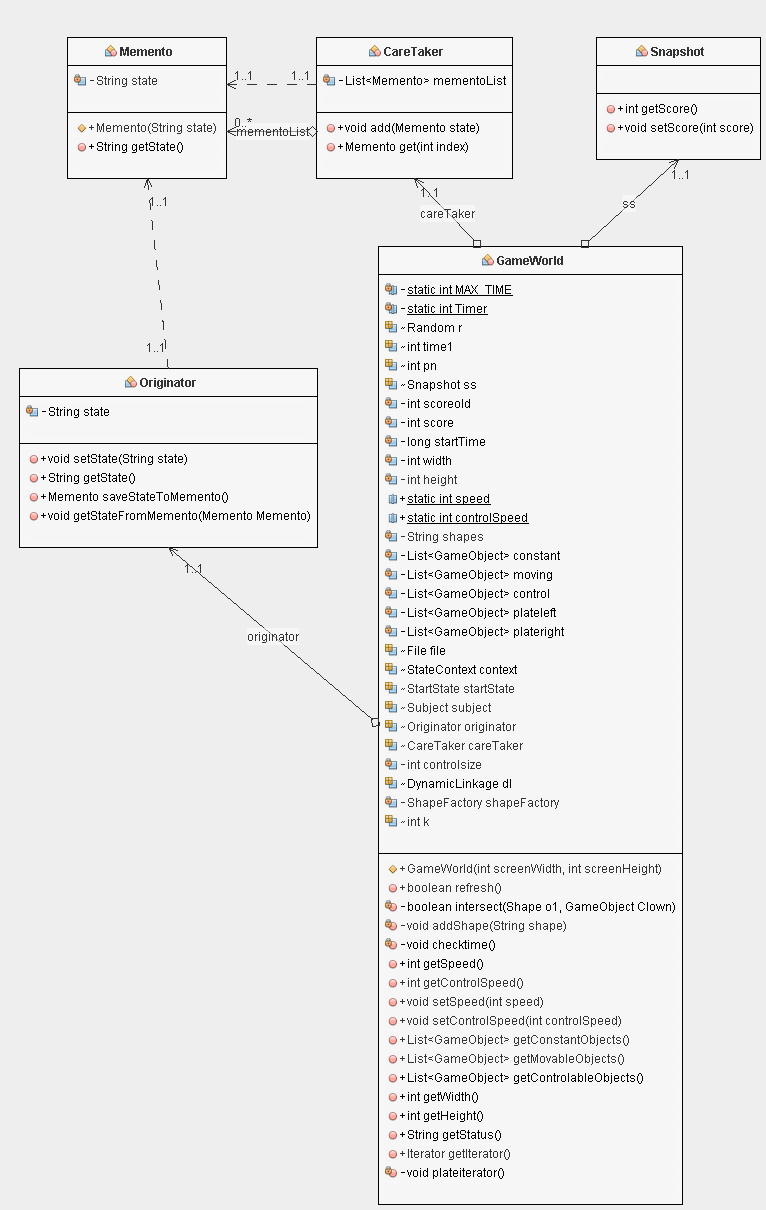
It reads from a file (dynamic linkage .txt)the names of the image file of the new added shape and their colors, Then it will send them through the dynamic factory that creates several objects of the selected shape and then sends the selected objects through the game world and then into the game itself.



5-snapshot :

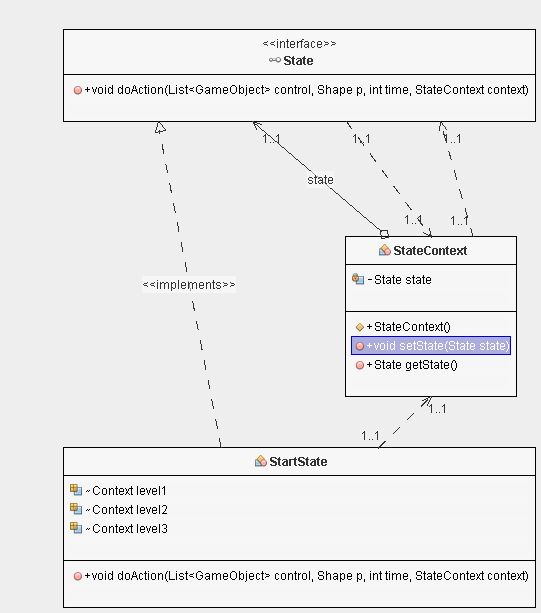
The snapshot pattern is used to restore the state of an object to a previous state , here when the user quits the game , the score (object) ‘s value is saved in a file along with all the game status like time and level.(memento.txt)so that it’s restored when the game resumes again.

The snapshot classes here are momento that takes status of the game and stroes it in an external file .



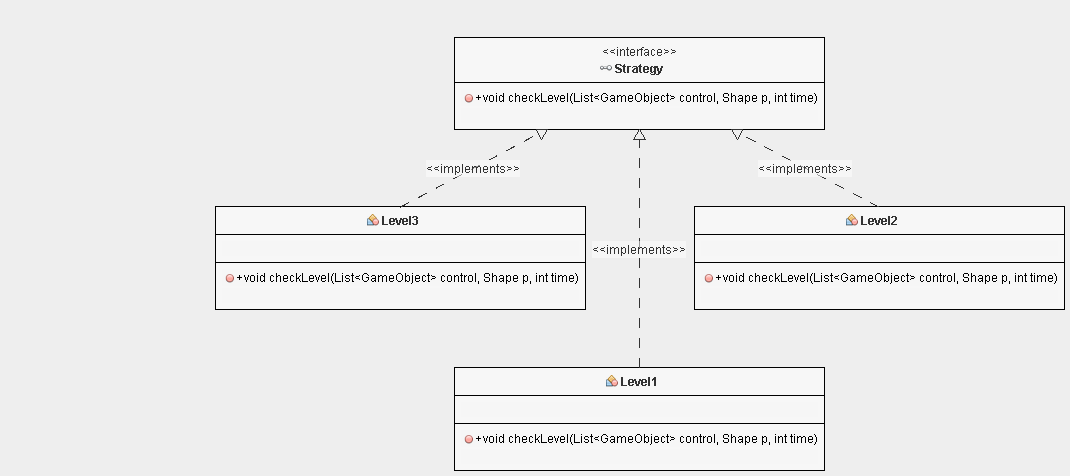
6- State :

The state pattern is a behavioral design pattern much like the strategy pattern, it creates objects that represnets various statues and context objects (the 3 levels ) that are changed according to the state context of the game.



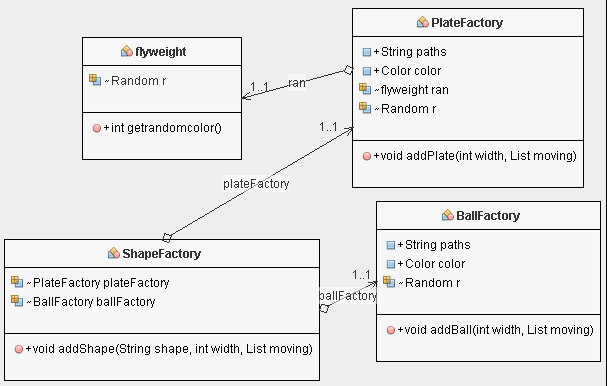
7- strategy :

In Strategy pattern, we create objects which represent various strategies and a context object whose behavior varies as per its strategy object. The strategy object changes the executing algorithm of the context object. It consists of the main strategy function that checks the level of the game currently from 1,2,3 , and the context function that executes from the strategy. And implements the levels 1 or 2 or 3 accordingly from the strategy.



8- flywegiht :

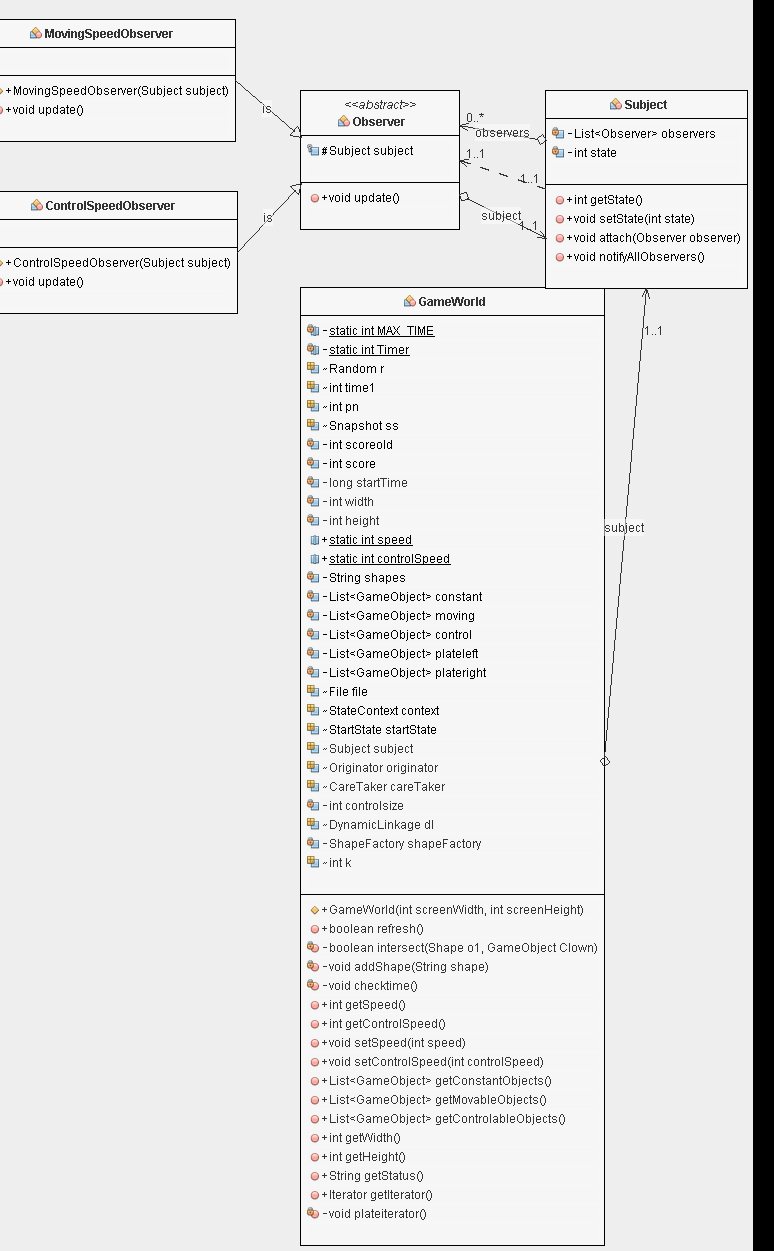
The fly weight pattern is used to reduce the number of objects created in order to decrease memory usage and eliminate redundancies. The flyweight class is respeonsible to get random colors, so it returns a random number in both the plate and ball factory classes.



9- Observer :

The observer pattern depends on a one to many relationship between objects , so if the object is modified , the objects depnding on it are automatically updated.

Here the subject class accuires the state of the score and when it chenges ( the score increases), Then sends the changes needed to the observer class that controls the moving speed of plates and the control speed of the clown itself.



10- Read only:

The read only design pattern depends on removing the setters and getters in a given class and then provide getters only , so we’ll have a read only object thus making it immutable.

Here the Dynamiclinkageint is the Read only interface that the user (dynamic factory) creates getter objects from that interface. The getters themselves are fount in the dynamic linkage class.

