Connect the gems

**Program Engineering Project**

**Contents:**

1. Introduction
2. What does the game do?
3. Structure of the game
4. Template definition
5. Highlighting templates
   1. Singleton template
   2. Observer template
   3. Factory Method template
   4. Prototype template
6. UML diagram
7. Bibliography
8. **Introduction**

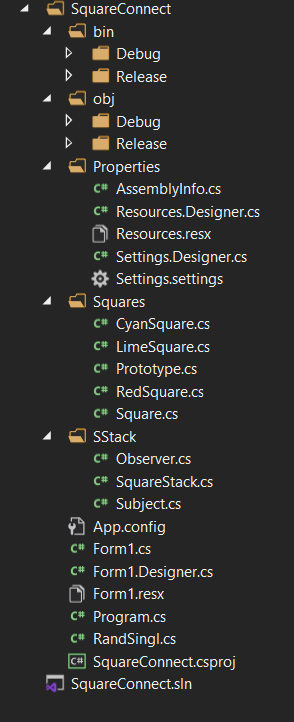
The project is realized having as inspiration the game "Connect the gems" which can be found at the following address: <https://www.puzzleplayground.com/g/connect-the-gems>

Visual Studio was used for this project and as programming language we used C#.

1. **What does the game do?**

As a base, this game has as its goal to connect as many carers of the same color, connecting them together and thus getting as many points. Inspired by the collection of games "Candy Crush", "Connect the gems" is a more simplistic and friendly game, predestined for beginners and enthusiasts who want to put their minds to work to get as many connections from as few moves and to be able to have a higher score.

1. **Game structure**



It has a rather simplistic structure with the main class declaring the components needed to create the graphical part and their functionality. After this follows different files in which are the templates used.

1. **Template definition**

In programming, templates are a feature that allow code to be written without regard to the type of data that will eventually be used. Templates are used as workarounds for software problems and to make programmers' work easier as well as to make a program easier to read.

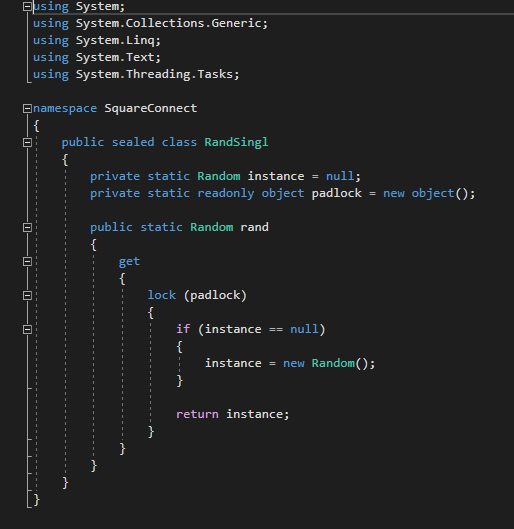
The advantage of templates is generally found in macro programs being safe in terms of consistency of the type of data used. Templates circumvent some of the errors commonly found in code that abuses macros.

1. **Highlighting templates**

In the following you will see a brief presentation of the templates used in this project, a brief presentation of it, how to apply it and how it improves our game (the real advantages of templates are observed on projects of a much larger size but I will present the theoretical advantages).

**5.a. Singleton template**

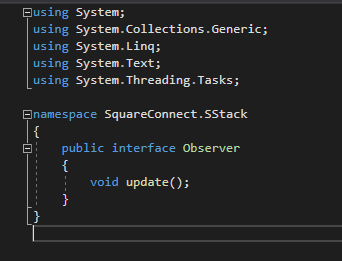
Singleton is a creative design pattern that allows you to ensure that a class has a single instance while providing a global access point to that instance..



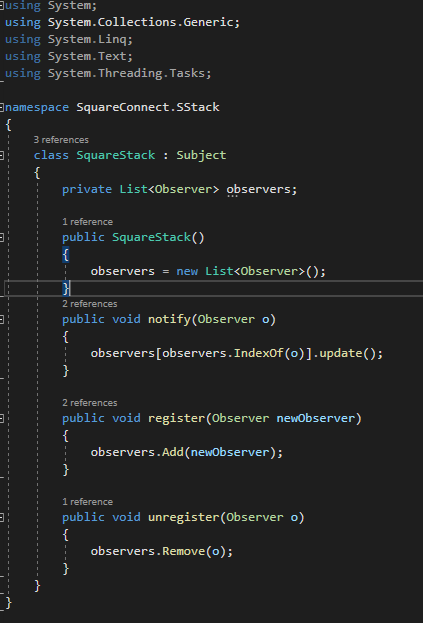
**This template has been used to instantiate the rand function. We declared this function once and later used it to return our instance.**

**This template is generally used for instantiating classes and is very useful in database connections so we can be sure that we can't call it again by mistake. At the same time this template can also reduce run times but it is visible in a much larger project**

**5.b. Observer template**

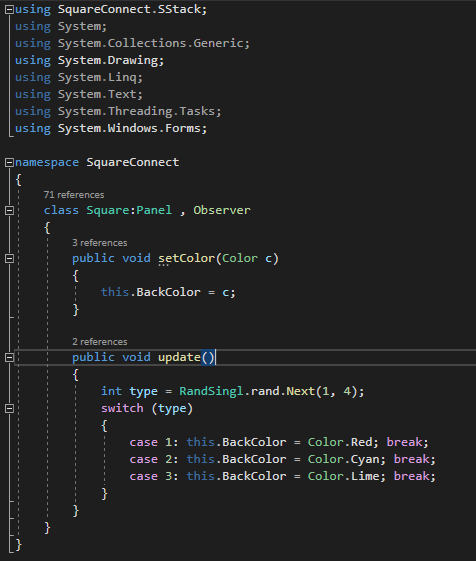
Observer is a behavioral design pattern that allows you to define a subscription mechanism to notify multiple objects about any events that happen to the object it is observing.

It facilitates communication between classes.Notify game areas to change their color when the validation condition to select the area that qualifies for scoring is met.

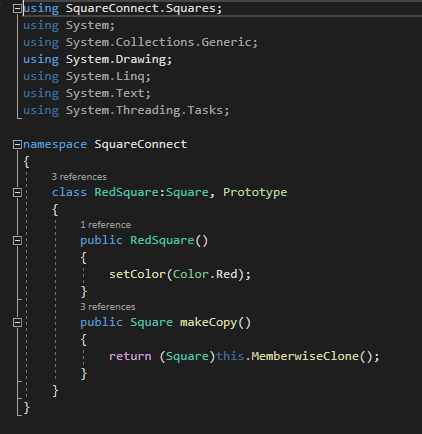


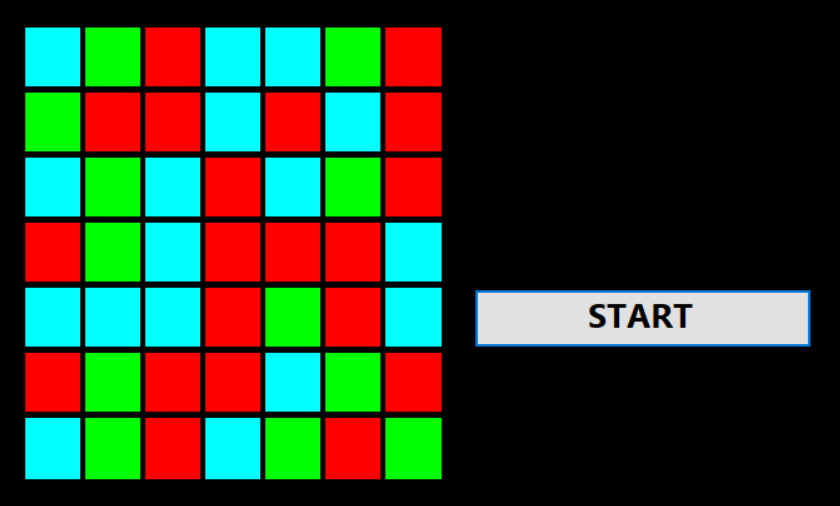
**5.c. Factory Method template**

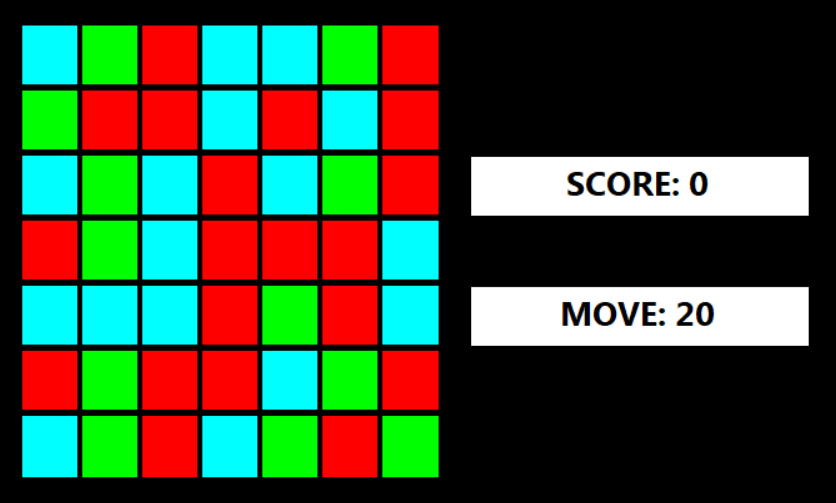
Factory Method is a creative design pattern that provides an interface for creating objects in a superclass, but allows subclasses to modify the type of objects to be created.

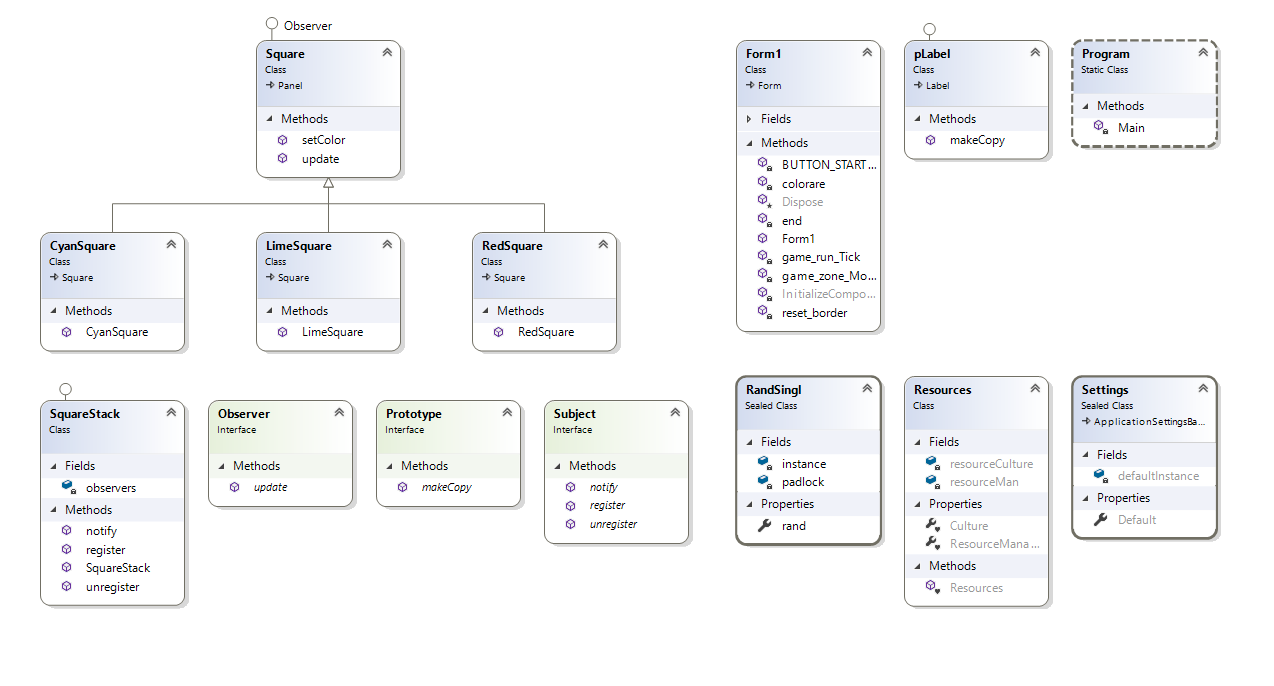
It organizes the game area system so I used the factory pattern to create using the superclass "square" through which I created the structure later this inheriting three subclasses: cyansquare, redsquare, limesquare; which I use the setColor method to customize them.

**This is a very useful template for graphical changes and settings giving a lot of freedom in its use. It also preserves the readability of the code. In our case it was used to change the background color.**







1. **UML diagram**
2. **Bibliography**

* https://refactoring.guru/design-patterns
* https://www.dofactory.com/net/factory-method-design-pattern
* https://www.dofactory.com/net/observer-design-pattern
* https://www.dofactory.com/net/decorator-design-pattern
* https://www.dofactory.com/net/builder-design-pattern