ESCAPE

**Software Engineering Project**

**Cuprins:**

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

The inspiration for the project is the game "Escape" which can be found at the following address: : <https://www.mathplayground.com/logic_40x_escape.html>

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

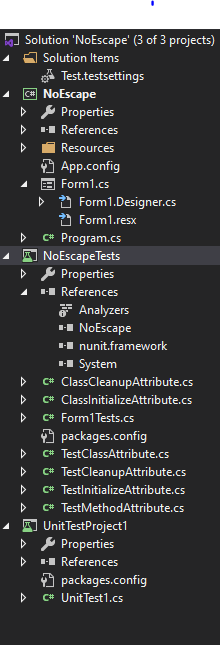
1. **What does the game do?**

The game is based on classic escape ideology with the main aim of using the player's agility, attention and deduction in an attempt to decipher the code to open the door.

The player must press the code in the correct order within a set time. Performing this operation once is not enough and with each advanced level the speed of the operation must increase as the time available to him becomes shorter.

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 various classes needed to realize the templates. .

****

Attached we have all the files present in this project:

* Tests settings
* Game
  + Properties
  + References
  + Resources
  + Code
  + Tests

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.

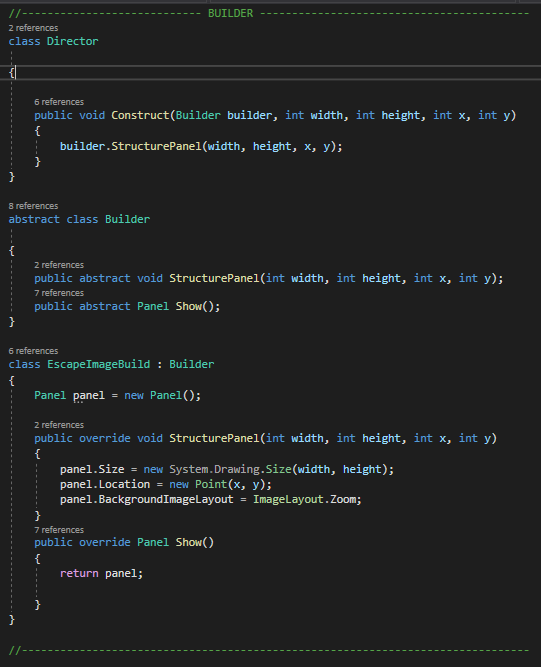
The advantage of templates is generally found in macro programs being safe in terms of consistency of the type of data used. Templates circumvent some of the errors commonly found in code that abuses macros.

**5.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. Builder Template**

**Builder is a creative design template that allows you to build complex objects step by step. The model allows you to produce different types and representations of an object using the same construction code.**



**To implement this template we used three classes.**

**The Director class is used to call the construction of the object which in turn calls the Builder class.**

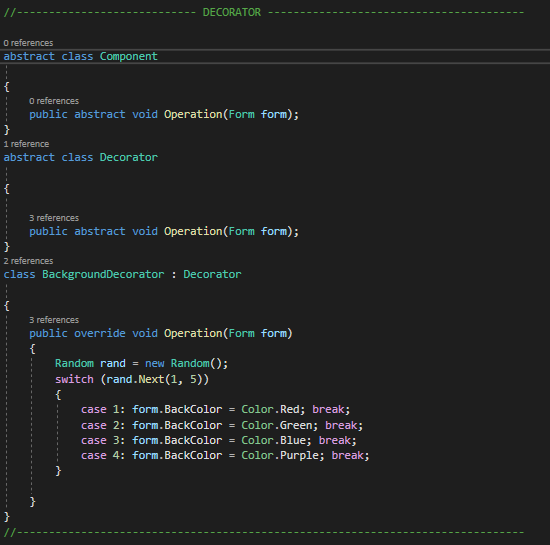
**In the Builder class we find the declarations of the functions called to construct and return the object.**

**EscapeImageBuild is the class where we find the functions mentioned above. Here we actually construct the object and return it.**

**This is a very useful template being able to be used to streamline the construction of objects thus greatly streamlining the working time and at the same time keeping the code much more readable. On a large project it can have a major impact. In our application it was used to create the "ESCAPE" writing elements which are separate elements where a change of images is used to highlight the player's progress. **

**5.b. Decorator template**

The decorator is a structural design pattern that allows you to attach new behaviors to objects by placing those objects in special wrapper objects that contain behaviors.

****

**For this template we used three classes.**

**The Decorator class is used for calling the object's modification this in turn calls the Decorator class and takes over the component to which the modifications are assigned.**

**BackroundDecorator is the class in which we find the function In which we randomly assign the graphical element, in this case the background color.**

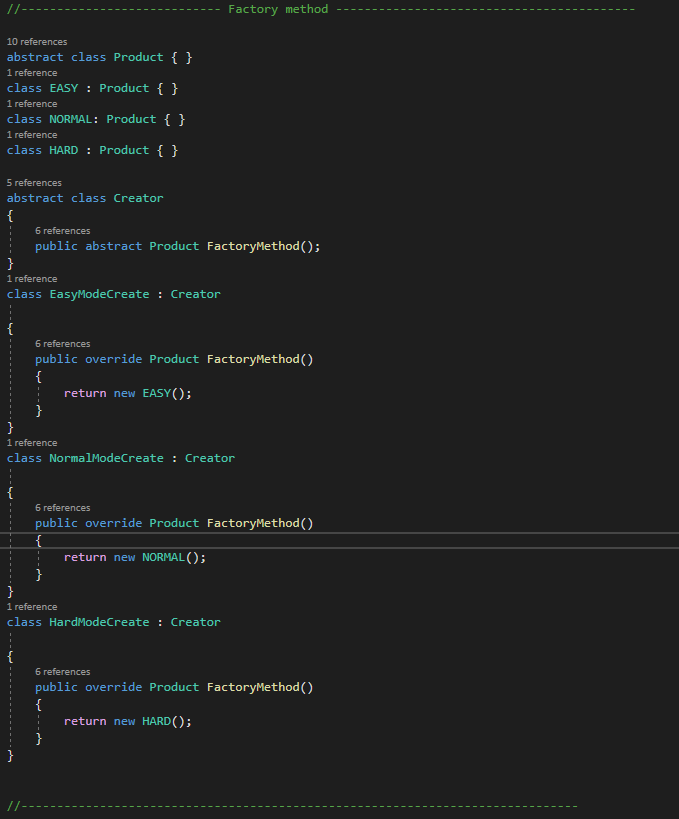
**This is a very useful template for graphical changes and settings giving a lot of freedom in its use. It also preserves the readability of the code. In our case it was used to change the background color.**

****



**5.c. Factory Method template**

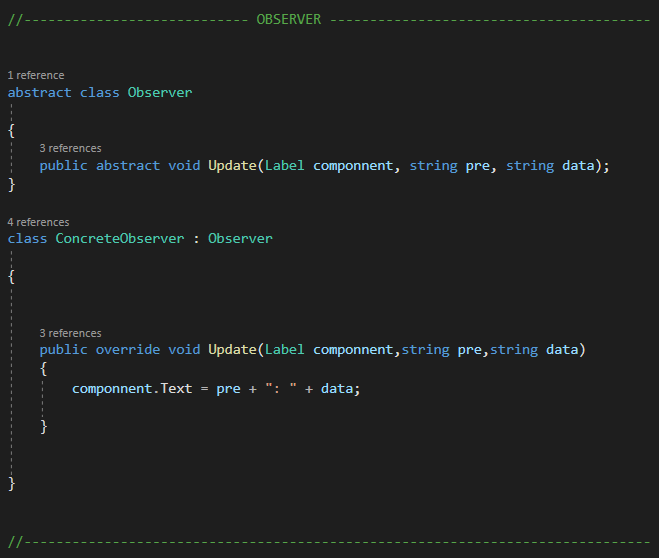
Factory Method is a creative design pattern that provides an interface for creating objects in a superclass, but allows subclasses to modify the type of objects to be created.



**Classes in this template are used to create different categories of objects. The three classes return another class representing the three difficulty categories in the game. In the program we only used these classes to retrieve the name that was assigned to the selection buttons.**

**It is mainly used to create multiple types from an object helping to optimize the code.**

**5.d. 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. 

**For this template we used two classes.**

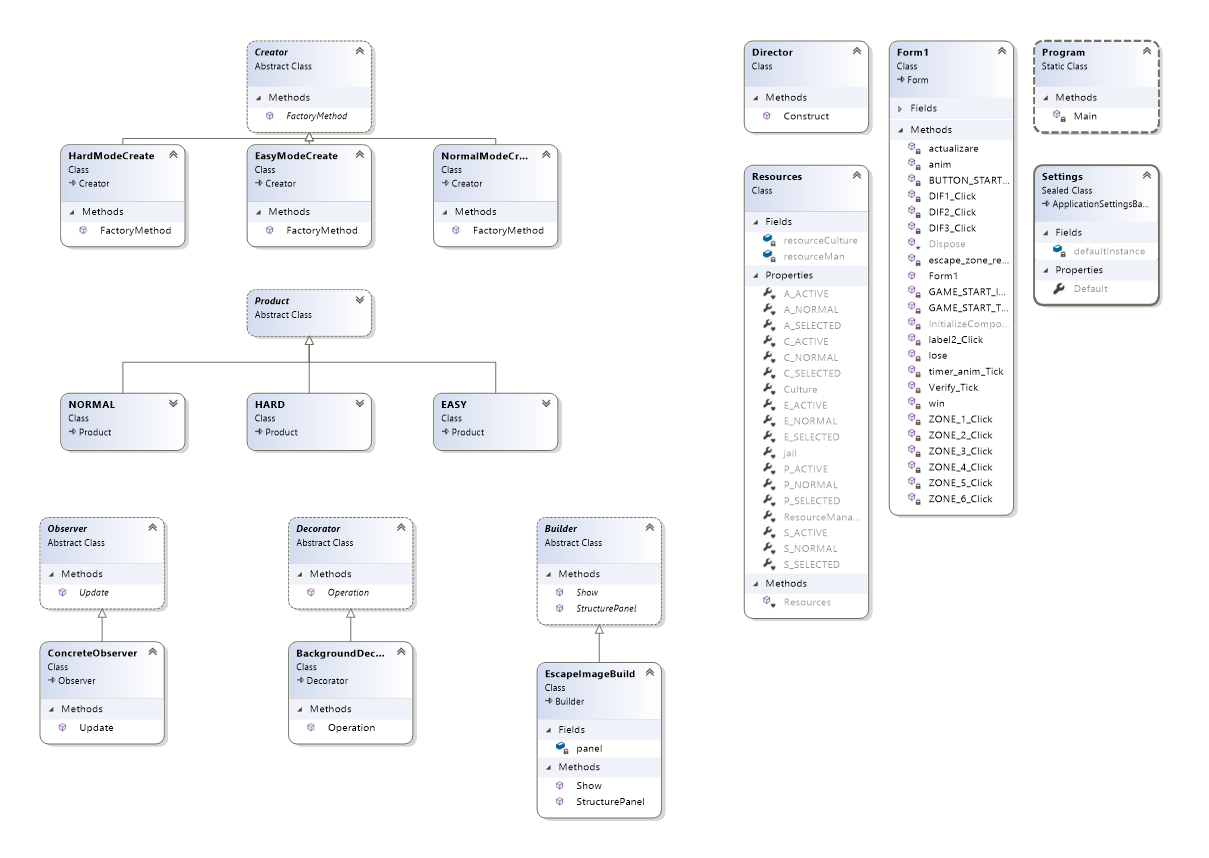
**The Observer class which calls the class that handles data updating.**

**Concrete Observer which is in charge of setting the text according to the received parameters thus keeping the game content updated to the game action.**

**This template is very useful to keep the data in the game up to date and control it in a very easy way.**



1. **UML diagram**



1. **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