PV260 Software Quality

Assignment 2 - Code Refactoring

Spring 2018

1 General Information

1.1 Dates

- Assignment start: 9.4.2018 (group 01), 30.4.2018 (group 02)
- Dry Run submission: 16.4.2018 23:59 (group 01), 7.5.2018 (group 02)
- Assignment deadline: 23.4.2018 23:59 (group 01), 14.5.2018 (group 02)

1.2 Submission

Please submit your solution as a .jar/.zip file containing only source files or a link to a Github repository to the Homework vault (Odevzdávarna) called *Assignment 2: Refactoring (Groups 01, 02)*. Name of the submitted file should be as follows: *lastname1-lastname2-assignment2.jar/zip/txt*. The tasks should be solved in the groups of two. Only one solution per group should be submitted.

1.3 Evaluation

The maximum points for this is assignment is 15. The points will be distributed based on both fulfilling the functional requirements and compliance of the code with SOLID and Clean Code principles. Additional points can be awarded for high quality solutions.

1.4 Project

This assignment works with the following project:

- TRON game: `http://www.mediafire.com/download/ucj53886o8a1164/Tron.zip`

2 Mandatory Tasks

The following tasks are not meant to be completed in any particular order.

2.1 Task 1

- Make it possible to add more players to TRON easily (so 3,4,5…people can play at the same time each with their own uniquely colored bike)
- Addition or removal of players should be as simple as possible, take into account the collision detection, controls configuration, rendering etc.
- Addition of player will be done offline, that is to change number of players the source of the game will be altered and the whole code recompiled. There is no need to create any kind of game menu.

2.2 Task 2

- Adapt current code to make it possible to use as much of it as possible as a generic engine for implementing other games.
- Provide a way for these new implementations to bind with the engine, e.g. interfaces that need to be implemented.
- Make the current TRON game such implementation of the engine.

2.3 Task 3

- Add the possibility to control any player by mouse (e.g. turn left or right using left and right mouse buttons respectively)

2.4 Task 4

- Separate the state of the game from its graphical representation, that is have one class / package represent the Model and other the Presentation.
- The model must contain and provide access for the player bikes, events such as collision between two players happened and so forth.
- There is no dependency from Model to Presentation, only from Presentation to Model
- The model itself should not handle frame timing (Thread.sleep()), that is the responsibility of calling some update method on the model is outside of the model, preferably in the extracted engine code from previous tasks.

3 Optional Tasks

3.1 Task 5

- Use the extracted engine to implement some other game (e.g. pong, snake,...)

3.2 Task 6

- Use git for your solution and structure your commits in a similar way to the demo/exercise, i.e. the commits represent the individual steps in the refactoring process