PROGRAMMING ASSIGNMENT 3

Issue Date : 19.04.2024 - Friday
Due Date : 12.05.2024 - Sunday (23:00)
Advisor: R.A. Aslı TAŞGETİREN
Programing Language : Java 8

1 Introduction

In this experiment you are going to gain experience in User Interface (UI) and game design in Java using JavaFX library.

2 HU-Load

HU-Load is a mining game, which was inspired from Motherload game, in which there is a machine that drills the earth to reach and collect various valuable minerals and gems, and these turn into money when they are mined/collected. Objective of this game is to collect as much money possible before machine's fuel runs out.

2.1 Game Mechanics

These rules, elements, and processes should be followed while developing the HU-Load game:

2.1.1 Elements

- The machine (i.e. the drill) has a fuel tank, storage and money case. It is controlled with arrow keys and it drills into underground.
- Underground there are different types of elements: earth (soil), valuables (minerals, gems), boulders and lava.
- Earth/soil can be plain or if they are at the top they can have grass. Drill can easily drill into these, it loses fuel while doing this but does not gain any money or weight.
- Valuables has weight and worth, these can be collected while drilling which allows drill to collect their worth as money, but it also collects their weight as well.
- Boulders are stones that cannot be drilled into, they can occur anywhere on the underground, but they must absolutely be present in the edges of the play area.
- Lava are harmful parts of underground, if machine ever drills into a lava spot it causes game over.

2.1.2 Rules and Processes

- Game must have at least three valuable types.
- There must be more soil than there is any other element in the game screen.
- Borders of the underground(except top) should be made up by boulders.
- Drilling into a lava destroys the drill which causes a game to be over.
- Drill is always running even when it is not drilling, so it should be always losing some small amount of fuel, however the significant loses happens while drilling.
- Losing all of the fuel also causes game to be over, when the game over is caused by fuel shortage, game over screen should also include total collected money.
- Machine should be facing the direction it is drilling, if it is flying then it should have its flight blades out.
- Machine can only fly up in empty spaces, it does not have ability to drill upwards.
- There is gravity in the game, if there is no game element under the machine, it should fall down.

3 Assets

You will not have sample input or output files in this assignment, however you will be provided with an assets file that will include necessary images for the game. Since all of you will be using the same assets, you will not submit these with your assignment. For consistency, write your code in a way that it assumes assets folder is in src folder.

4 Checklist

Along with this introduction pdf you are also given a Checklist file, this file includes list of all of the tasks that your game should be doing at the bare minimum. If you believe your game meets requirements of the item on the list you should mark it done, every item that was checked in the list should be shown in the demo video. Dishonesty will lose you points. You should include the link to your demo video at the end of this list. Additionally, in this assignment you should also provide UML Class Diagram of your project, you can add this to the end of the Checklist file. You will submit this file in pdf format.

5 Demo Video

In this assignment you will also include a demo video that shows all the requirements of your assignment, your demo should show every task you have marked done in your checklist and it should also include a vocal narration and explanation of how these work.

Your video should be no more than 4 minutes. If it is longer than that limit you will lose points depending on how much time it is past the limit.

The video should not be open to public, you can ensure this by uploading your video to Youtube as an unlisted video i.e. a video that can only be seen by those who have the link.

6 Extras

Optionally you can include some extra capabilities in your game, these will allow you some bonus points. One of the extras can be vertical scroll [1]. The checklist has some empty spaces to allow you to enter these extras. **Make sure you first do the minimum requirements before you attempt to do any extra work.**

Execution and Test

Your code should be compiled and run as such:

- Compilig: javac8 *.java, or javac8 Main.java
- Running: java8 Main

Grading Policy

Task	**Point**
Checklist Tasks and Demo Video	100*
Total	100

***Just submitting these are not enough, you will be graded by how well you have done the tasks, how well done your demo was and how good your code was. As always you should follow the rules stated in this pdf and in Piazza, otherwise you will face point deductions.**

Submit Format

File hierarchy must be zipped before submitted (Not .rar, only .zip files are supported by the system)

```
- b<studentid>.zip
    - Checklist.pdf
    - <src>
       - Main.java, *.java
```

Late Policy

You have two days for late submission. You will lose 10 points from maximum evaluation score for each day (your submitted study will be evaluated over 90 and 80 for each late submission day). You have to submit your solution in deadline date + two days, otherwise it will not be evaluated.

Notes and Restrictions

- **You will be developing this game with pure JavaFx, you are not to use other tools such as Java Swing or Java AWT or FXML etc.**
- Do not miss the submission deadline.
- You must obey given submit hierarchy and get score (1 point) from the submit system, if not as stated in BBM 104 Laboratory Rudiments you will lose 10% of your grade.
- Save all your work until the assignment is graded.
- Source code readability is a great of importance for us. Thus, write READABLE SOURCE CODE, comments and clear MAIN function. This expectation will be graded as "clean code".
- Regardless of the length, use UNDERSTANDABLE names to your variables, classes and functions. The names of classes, attributes and methods should obey Java naming convention. This expectation will be graded as "coding standards".
- You must use JavaDoc commenting style for this project. Use comments to explain design and algorithm choices you have made in your project. There is no need to state the obvious, (e.g, a variable named price is price of something), be brief, follow coding standards and write clean code.
- You can ask your questions through course's piazza group and you are supposed to be aware of everything discussed in the piazza group. General discussion of the problem is allowed, but DO NOT SHARE answers, algorithms, source codes and reports.
- All assignments must be original, individual work. Duplicate or very similar assignments are both going to be considered as cheating.

References

[1] https://en.wikipedia.org/wiki/Vertically_scrolling_video_game

[2] https://docs.oracle.com/javase/8/javafx/api/javafx/application/Application.html

[3] https://docs.oracle.com/javase/8/javafx/api/javafx/stage/Stage.html

[4] https://docs.oracle.com/javase/8/javafx/api/javafx/scene/Scene.html

[5] https://docs.oracle.com/javase/8/javafx/api/javafx/scene/layout/package-frame.html

[6] https://docs.oracle.com/javase/8/javafx/api/javafx/scene/canvas/Canvas.html

[7] https://docs.oracle.com/javase/8/javafx/api/javafx/scene/paint/Color.html

[8] https://docs.oracle.com/javase/8/javafx/api/javafx/scene/input/class-use/KeyCode.html

[9] https://docs.oracle.com/javase/8/javafx/api/javafx/animation/Animation.html

[10] https://docs.oracle.com/javase/8/javafx/api/javafx/scene/text/Text.html