Computer Science E214 (2020)

Project: Star Line

Group number:
Group members:

Member #1: Jacques Wüst 22794425
Member #2: Andries Phillipus Lotriet 21617066
Member #3: Ryno Slabber 22633456

Table of Contents

Execution Details

Start in the GlowLine-main folder that contains the src folder.

The copy command after compilation is there so that the game can access the CSS file. Otherwise, the GUI loses all styling.

Linux:

javac -classpath src/ -d ./ src/MainGame/MainGame.java

cp src/MainGame/Layout/GLM1080.css MainGame/Layout/

java -classpath ./ MainGame.MainGame

Windows(cmd):

javac -classpath src\ -d .\ src\MainGame\MainGame.java

copy src\MainGame\Layout\GLM1080.css MainGame\Layout\

java -classpath .\ MainGame.MainGame

Interface Inheritance

Main game loop: The main game loop implements the handle() method form StatusTime(), a custom class that is an implementation of AnimationTimer.

Collision handling: The abstract GameObject class specifies a collisionHandling() method which is in turn implemented by all game objects e.g Player, Lander, etc.

Class Inheritance

The GameObject class is the base class for all game objects such as Player, Lander, Bullet etc. GameObject methods that all game objects require, update(), accelerate(), etc. and variables that all game objects require, collisionShape, type, dead, etc. are all defined in GameObject.

Summary of Additional Work

- Free movement from level 2: from level 2 the user is not confined to moving left and right. Instead, the user can accelerate the player in any direction with the mouse and W-key.
- Particle system: we created a particle system for explosions, exhaust plume, etc. to greatly improve graphics.
- Smart-tracking missile enemies: on level 2, enemies appear that will track the player in an intelligent manner, i.e. they use the player's current relative velocity to calculate an intercept vector and then move accordingly. These enemies explode on contact with the player.
- Accurate collisions: all collisions are accurate to the actual shape of the objects. When the player impacts the floor this is easy to see.
- Intelligent object-out-of-screen handling: the player, missile enemies and bullets teleport to the other side of the screen when exiting the screen on one side. The player bounces off the ceiling. The player stays above the floor.
- Time dilation power: this power-up slows downtime. Use the SPACEBAR during gameplay to experience this power-up. The most difficult part of implementing this feature was limiting how much the user can use this power-up in an intelligent way and displaying the usage.
- Increasing difficulty: while there are only two levels, the second level gets increasingly more difficult by increasing the speed of enemies, making strategic use of the time dilation power-up important to attain a higher score.
- Custom difficulty: although the difficulty will always increase in level 2, the rate at which it increases and the difficulty at which the game starts, can be set by accessing the "Controls & Settings" button and using the slider.
- High-score system: the score of all games are saved in a text file and displayed on the main menu in descending order. Note that you can scroll the high scores on the main menu to see all of them. They also display the percentage difficulty at which the score was obtained.
- Music & sound: Star-Wars-themed music and sound effects were added. Note that while the audio itself was obtained from YouTube, all of the files were manipulated with Audacity to get them perfect.
- Dynamic scale: as the game is designed to be used full-screen, the game might be played on different resolutions. This will cause severe scaling and object location errors with games of set sizes, but ours gets the resolution of the display, sets the game to that resolution, and places objects accordingly. The game was tested down to 1360x768
- Pause menu: a pause menu was added that can be accessed any time during the game by pressing the ESCAPE-key. Gameplay will resume normally when the user clicks on the resume button or if the user presses the ESCAPE-key again.

- Improved graphics: advanced custom-animated swelling, unfolding buttons were designed for the game, great care was taken to reimagine the classic game's graphics in a futuristic context.
- Intelligent turret: instead of the turret rotating with lousy keyboard keys, the direction of the turret to the mouse is calculated and set. This makes the advanced play mechanisms in level 2 possible.
- Extra lives: the user has 3 extra lives. The lives are indicated with heart shapes. They disappear as the user loses lives. Note that the SVG info for the heart shapes was obtained from the web.
- A timer is shown in-game.

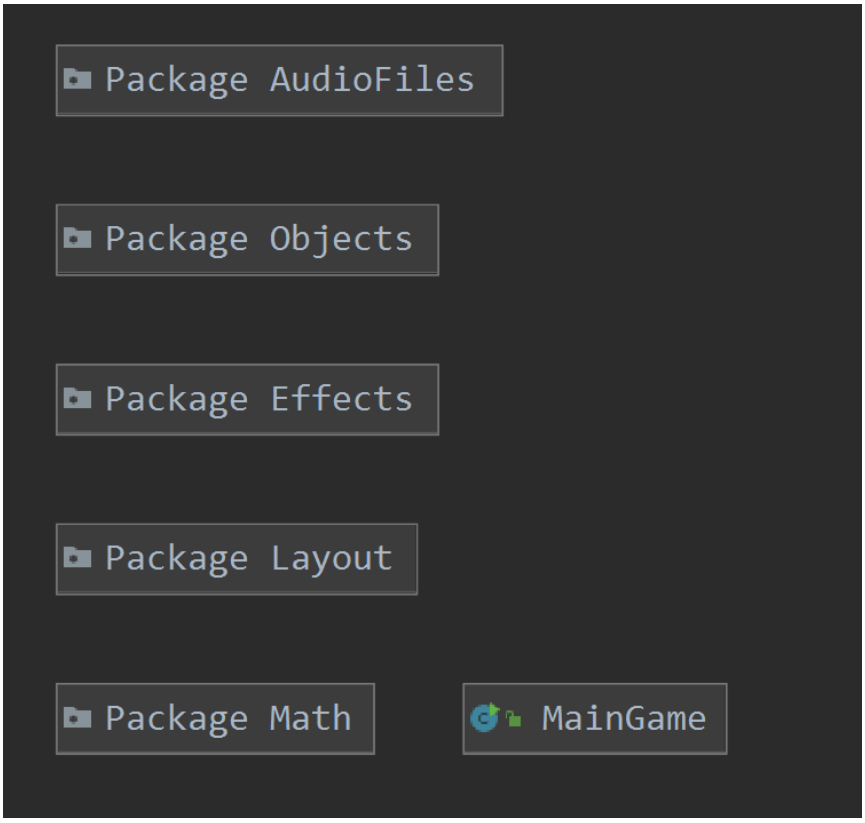
External Libraries

While no external libraries are used in this project, stdlib is not used either. The project is built using the internal javafx library.

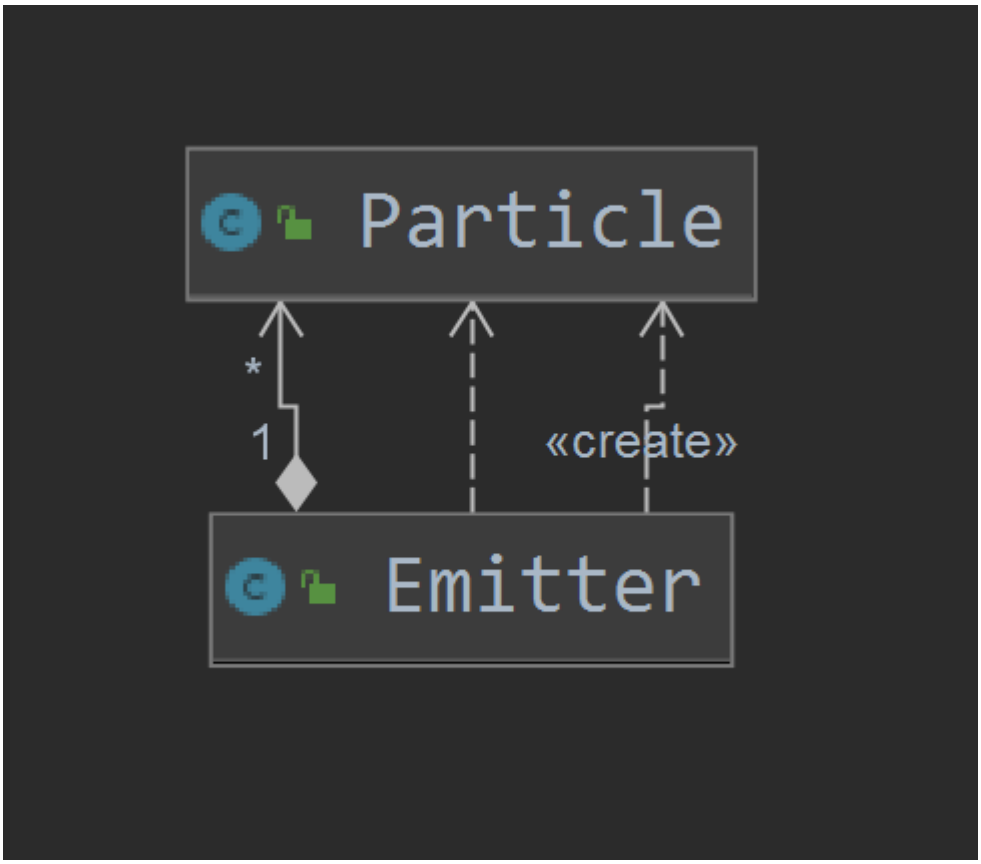
We decided to use javafx because it provided a way to create a true full-screen application (although to change the resolution of the monitor swing is also used), and to facilitate more advanced graphics without lagging.

Class Structure Diagrams

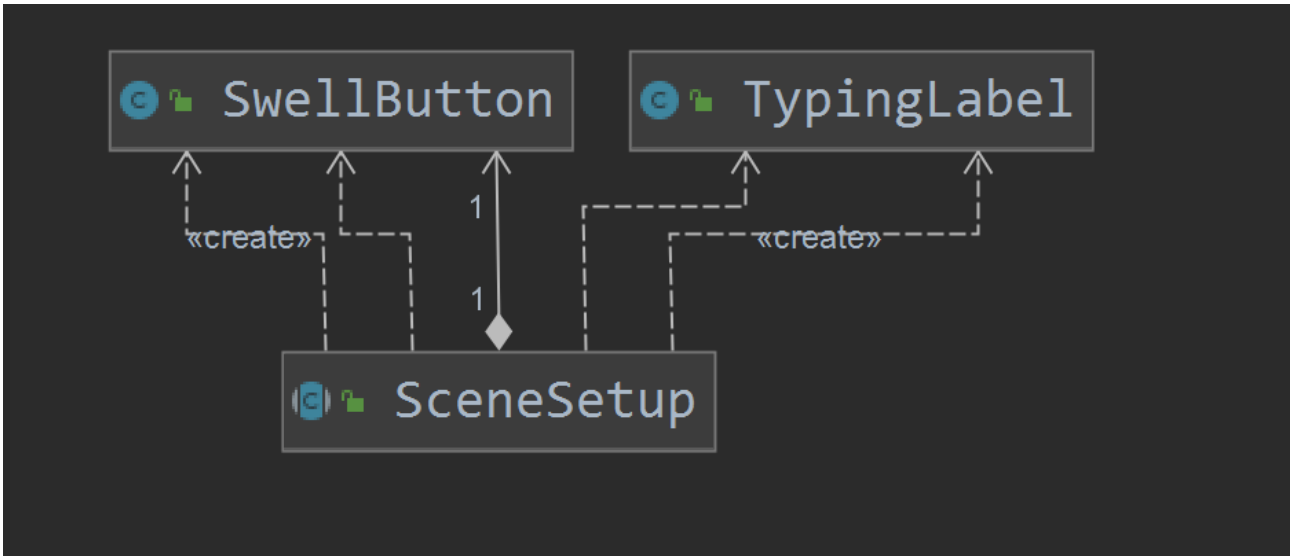
MainGame:



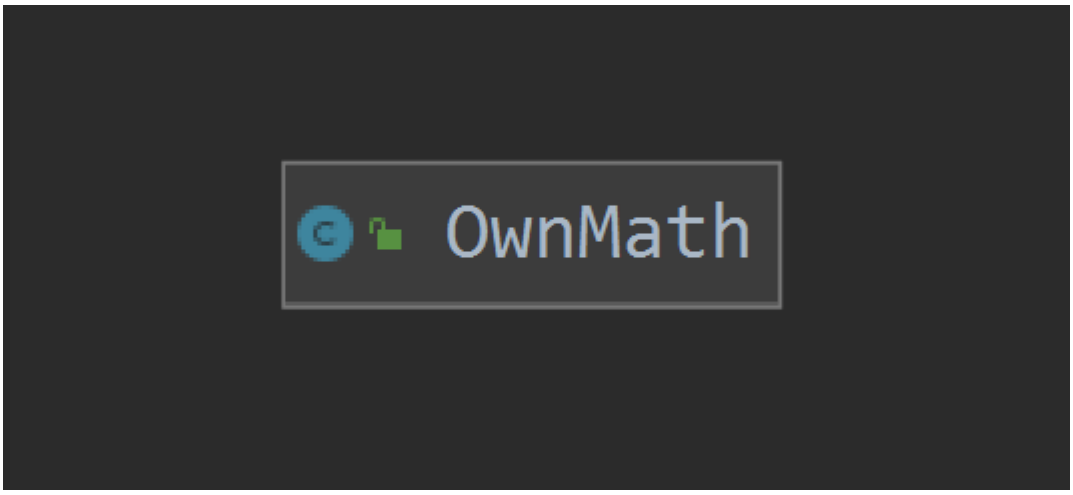
Effects:



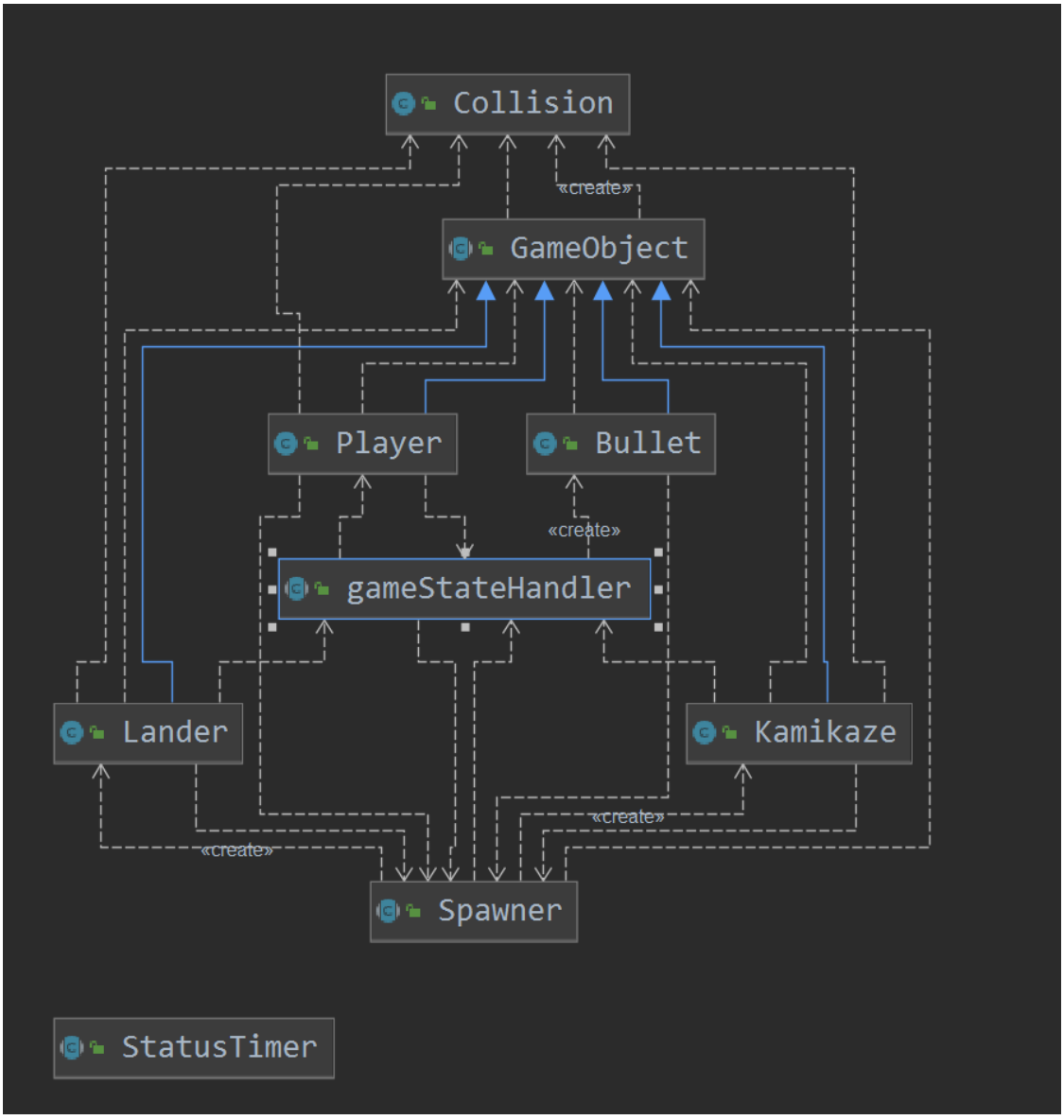
Layout:



Math:



Objects:



External Sources

KeyListener : https://stackoverflow.com/questions/29962395/how-to-write-a-keylistener-for-javafx

ScreenResolution : https://coderanch.com/t/503348/java/set-screen-resolution

Run FX in AWT Window: https://docs.oracle.com/javase/8/javafx/api/javafx/embed/swing/JFXPanel.html

Multi input method: https://stackoverflow.com/questions/2330942/java-variable-number-or-arguments-for-a-method

Canvas doc: https://docs.oracle.com/javase/8/javafx/api/javafx/scene/canvas/Canvas.html#method.summary

Graphics context doc: https://docs.oracle.com/javase/8/javafx/api/javafx/scene/canvas/GraphicsContext.html

Particles 1: https://natureofcode.com/book/chapter-4-particle-systems/

Particles 2: https://www.youtube.com/watch?v=vLcJRm6Y72U

Abstract: https://www.w3schools.com/java/java_abstract.asp

Enclosing class: https://stackoverflow.com/questions/20252727/is-not-an-enclosing-class-java

Iterators: https://www.geeksforgeeks.org/iterators-in-java/

GUI 1: https://www.youtube.com/playlist?list=PL6gx4Cwl9DGBzfXLWLSYVy8EbTdpGbUIG

GUI 2: https://docs.oracle.com/javafx/2/layout/builtin_layouts.htm

Pane: https://docs.oracle.com/javase/8/javafx/api/javafx/scene/layout/Pane.html

CSS 1: https://docs.oracle.com/javafx/2/css_tutorial/jfxpub-css_tutorial.htm

CSS 2: https://www.w3schools.com/css/default.asp

CSS 3: https://docs.oracle.com/javase/8/javafx/api/javafx/scene/doc-files/cssref.html#typecolor

Transition: https://docs.oracle.com/javafx/2/api/javafx/animation/Transition.html

Main menu background: https://wallpaperplay.com/walls/full/1/e/6/124127.jpg

Sound:

- https://www.youtube.com/watch?v=PpRVyBbBD5k
- https://www.youtube.com/watch?v=3LjGbxCi3vk
- https://www.youtube.com/watch?v=s3SZ5sIMY6o
- https://www.youtube.com/watch?v=EsvfptdFXf4
- https://www.youtube.com/watch?v=Rcw_vTx_0IA
- https://www.youtube.com/watch?v=C8VnCcg1t7w
- https://www.youtube.com/watch?v=9FHw2aItRlw