Modeling and Programming 2018-1: Tenth lab practice

Luis Daniel Aragon Bermudez 416041271

December 18th, 2017

Contents

Java Swing Demos: Drawing curves with lines, Chessboard and Caterpillar

This project contains four Java Swing Demos that use drawLine, fillRect and fillOval in order to draw the figures described by the practice's specifications. Each demo corresponds to one of the practice's exercises.

It uses some custom tools in order to make drawing using mathematical transformations easier. All the demos inherit from an abstract subclass of `JPanel` (`Demo`) which implements a method that returns an `ImmutablePoint2D` object that contains the coordinates of the center of the panel.

The `Demo` class also provides the specification of the method `getOrigin()` which provides each demo a way to customize the desired cartesian coordinate origin it should work with.

Drawing curves with lines

Creating the impression of curved lines using only straight sections relies on carefully translating the end points of the lines through each iteration. On the first quadrant and supposing the first line extends upwards from the origin, the shift that helps us simulate this effect corresponds to the following operations:

- Move the start point `delta` units to the right.
- Move the end point `delta` units down.

where `delta` is calculated by dividing the length of the axes by the number of lines to be used. The results of this transformation

The first and second exercises heavily relied on the `ImmutablePoint2D` in order to navigate and draw the lines.

Chessboard

Caterpillar

Building and running the program

The program can be built using gradle, the most common tasks are described bellow, for a full list of available tasks use the *tasks* task. If you're on Linux or Mac then running the following command from the project's main directory will be enough to build and run the program: `./gradlew run`. If you're windows use `gradlew.bat run` instead.

Some of the most common tasks are:

1. `./gradlew build`, compiles the program to `out/jar/Dog.jar`.
2. `./gradlew javadoc`, generates the program's documentation and puts it inside `doc/`.
3. `./gradlew run`, compiles the program, creates the jar and runs the application.
4. `./gradlew clean`, deletes all files and folders generated during the build process (except the .gradle directory).

Bibliography

- "Operating Systems: Three Easy Pieces (Chapter 7: Scheduling Introduction)" - *Arpaci-Dusseau*, 2014
- "Semaphore (programming)" - *Wikipedia*, 2017
- "The Little Book of Semaphores" - *Downey, Allen B.*, 2016
- "Over de sequentialiteit van procesbeschrijvingen" - *Dijkstra, Edsger W.* 1962 or 1963
- "Dijkstra's algorithm" - *Wikipedia*, 2017
- "State Design Pattern" - *SourceMaking*, 2017

Acknowledgements

For more information on the tools used to build, create and run this program refer to the following links:

- Apache Ant was used to create the build script.
- Python was used for the execution script.
- JetBrains' IntelliJ IDEA was used as the primary editor.
- Graph Online was used to create the directed graph G.