Prof. Dr. Oscar Nierstrasz
Pascal Gadient, Mohammadreza Hazhirpasand

Assignment 07 — 03.11.2021 – v1.0
Liveness and Asynchrony

Exercise 1 (2.5 Points)

Answer the following questions about liveness and asynchrony:

a) When should you consider using asynchronous invocations?

b) In what sense can a direct invocation be asynchronous?

c) What is an "early reply"?

d) What are "futures"?

e) When are futures better than early replies?

Exercise 2 (2.5 Points)

Answer the following questions about Java VM and implementation details:

a) Why does the call *Thread.currentThread().join()* not make much sense in a thread's concurrent run method?

b) Why can you encounter an IllegalMonitorStateException when calling notifyAll() in a code block that is not synchronized?

c) What happens with the thread when the code execution gets to the end of the thread's run method (suppose no loop is involved in the run method)?

d) Why do some Java apps not terminate, even though their GUI has been closed, and how can you mitigate that problem?

e) Name one reason for using a while loop in a thread's run method. Justify your selected reason.

Please continue on the next page!

Exercise 3 (5 Points)

The winter is coming and so is the snow. We created a thread-based snowflake environment to simulate winter conditions. In this setup each snowflake represents one thread. Consequently, each snowflake draws itself on the Java Graphics2D object from the main application. Your task is to synchronize threads and beautify the snowflakes and their movement. We provide several tasks with importance of their order:

- Make the snowflake a runnable thread.
- Instantiate some snowflakes and draw them!
- Movement of the snowflakes has to be synchronized. If it wouldn't be synchronized, each snowflake would just move so fast, you wouldn't even see any movement at all. In other words, they should fall to the ground each with a constant velocity.
- Java2D draw calls on the Graphics2D object have to be synchronized, because this measure greatly reduces flickering. It can be handled in several different ways, but the easiest one is probably to combine the draw call synchronisation with the movement synchronisation.
- Add individual horizontal and vertical movement parameters to the snowflakes.

Some things that don't matter and thus are not mandatory for this exercise (however, you can still work on these):

- Snowflakes themselves don't need to rotate
- Overlapping is not an issue
- Different sizes of the snowflakes are not required

You will find additional comments and hints in the code that may help and guide you. The SCG Snowflake Simulation Environment is available on GitHub. You can find the project on `https://github.com/pgadient/concurrency_e07t03.git`. Please submit your project within a zip file. You can easily achieve that by exporting the project with the Eclipse export assistant.