Concurrency:
State Models & Design Patterns

Practical Session

Week 10

Assignment 09

Discussion

A09 - Exercise 1

Answer the following questions:

a) What criteria might you use to prioritize threads (list at least 5 different criteria)?

- Representations of task priority, cost, price, or urgency.
- The number of tasks waiting for some condition.
- The time at which each task is added to a queue.
- Expected duration or time to completion of each task.
- Termination dependencies among tasks.

A09 - Exercise 1

Answer the following questions:

b) What are different possible definitions of fairness (list at least 3 different definitions)?

- **Weak fairness**: If a process continuously makes a request, eventually it will be granted.
- **Strong fairness**: If a process makes a request infinitely often, eventually it will be granted.
- **FIFO** (First-in, First out): If a process makes a request, it will be granted before that of any process making a later request.

A09 - Exercise 1

Answer the following questions:

c) What are Pass-Throughs and Lock-Splitting?

- **Pass-Throughs**: The host maintains a set of immutable references to helper objects and simply relays all messages to them within unsynchronized methods.
- **Lock-Splitting**: Instead of splitting the class, split the synchronization locks associated with subsets of the state.

A09 - Exercise 1

Answer the following questions:

d) When should you consider using optimistic methods (list at least 3 different enablers)?

- Clients can tolerate either failure or retries.
- You can avoid or cope with livelock.
- You can undo actions performed before failure checks

A09 - Exercise 2

In this exercise you have to **implement** a class that represents graphical objects that consist of an x-coordinate, a y-coordinate, a width and a height (**= rectangle**). The class has to implement methods for:

- Increase the x-coordinate by 10% and decrease the y-coordinate by 20% (change position)
- Increase the width by 50% and decrease the height by 70% (change dimension)
- Increase the y-coordinate by 40% and decrease the height by 60% (change position and dimension)

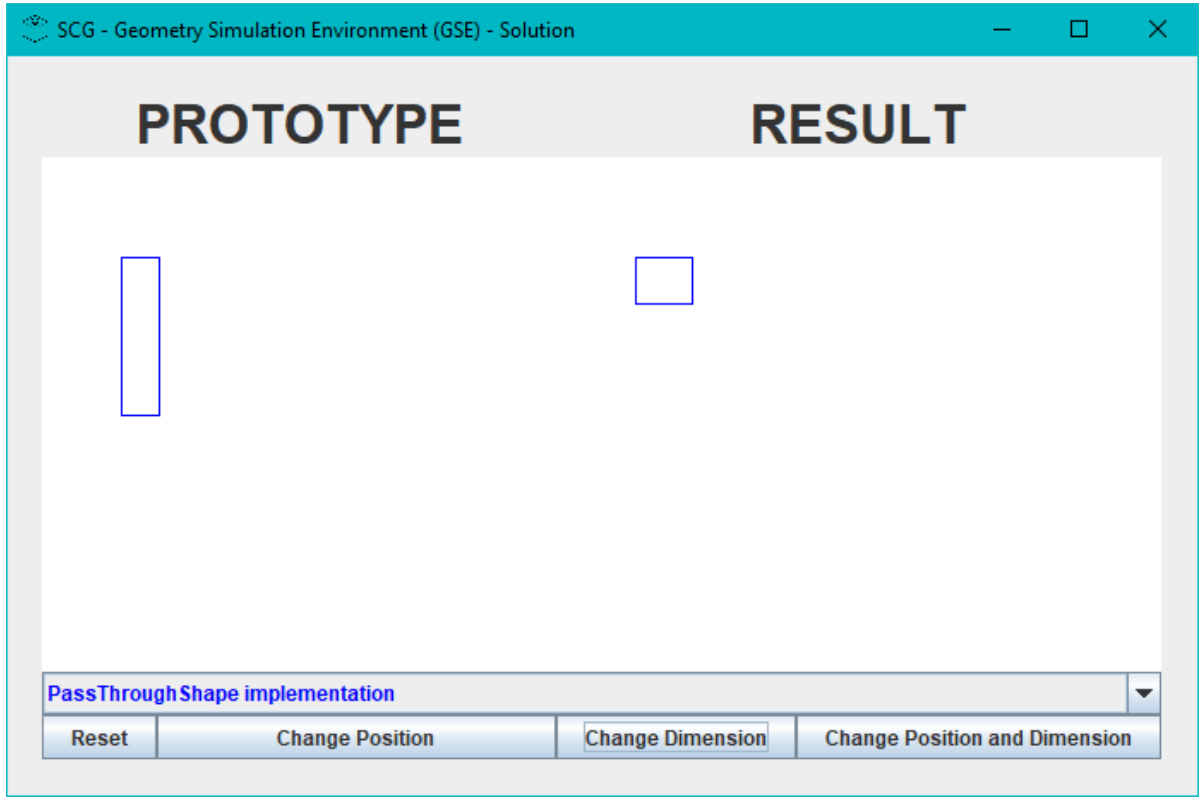
Implement it once using Lock-Splitting and once using Pass-Throughs.

A09 - Exercise 2

SCG Geometry Simulation Environment

(SCG GSE)

solution available on GitHub



A09 - Exercise 3

Answer the following general questions:

a) How do threads waiting in a Thread.join() loop get aware of that thread's termination?

All threads will call this.notifyAll() before they enter the TERMINATED state. This call is issued by the Java framework itself and therefore it is not visible in the code.

b) How could you optimize the code below?

```
Thread t = new Thread(<insert your runnable code here>)
t.start()
t.join()
```

You could remove the thread instantiation and extract the plain code into an ordinary method without any asynchronity.

A09 - Exercise 3

Answer the following general questions:

c) Are String objects in Java mutable or immutable? Justify your answer!

According to JavaDoc: The String class is immutable, so that once it is created a String object cannot be changed.

d) Does the FSP progress property below enforce fairness? Justify your answer!

```
progress HeadsOrTales = {head, tale}
```

No, it does not. When a process will choose head in every run it doesn't violate this progress property, but it won't be fair.

Lab 02

Details

Lab 02 - Overview

Concept

- Eclipse + Java based tasks

- Concurrency code that needs work

- Work in groups of two

- The lab starts at 10:15 and ends at 12:00

- Location: same as lecture

Grading

- 5 bonus points for 100% correctness

Requirements

- Notebook with power adapter

- Latest version of Eclipse installed

- Latest version of the Java SDK installed

- Optional: a mouse

Allowed:

- Lecture slides and books

- Internet access

Lab 02 - Workflow

1) Pull from public GitHub repository
   https://github.com/pgadient/concurrency_lab02.git
2) Fix the four provided sample apps
   - *Nested monitor*
   - *Concurrent read/write access*
   - *Fairness*
   - *Concurrent resource allocation*
3) Submit your zipped project solutions by mail to pascal.gadient@inf.unibe.ch

Lab 02 - Solutions

Pull from public GitHub repository

https://github.com/pgadient/concurrency_lab02_solution.git

Please keep in mind that I will turn this repository for obvious reasons into a private one by the end of the day!

So please hurry to retrieve your own copy.

We don't have any new exercises for you this week!