Prof. Dr. A. Poetzsch-Heffter
Mathias Weber, M.Sc.

University of Kaiserslautern
Department of Computer Science
Software Technology Group

Advanced Aspects of Object-Oriented Programming (SS 2013)

Practice Sheet 11

Date of Issue: 25.06.13
Deadline: 02.07.13
(before the lecture as PDF via E-Mail)

Exercise 1 Asynchronous vs. synchronous Calls

```
class A {
   C c;
   B b;
   void m1() {
      c.m(5);
      b.m2();
   }
}
```

```
class B {
   C c;
   void m2() {
      c.m(9);
   }
}
```

```
class C {
   void m(int i) {
      System.out.println("i=" + i);
   }
}
```

For this exercise, we assume that standard Java has been extended with asynchronous method calls and the programmer can choose between them.

a) Consider the given classes. What is the output of the following code in case of synchronous method calls and what happens if all calls are asynchronous? Why?

```
C c = new C();
A a = new A();
a.c = c;
B b = new B();
a.b = b;
b.c = c;
a.m1();
```

b) Compare asynchronous and synchronous communication between objects in a multi-threaded program context. What are the advantages and disadvantages of each communication technique?

Exercise 2 JCoBox-Chatsystem

We come back to the chatserver of exercise 10.2. This time we want to implement the server using JCoBox. You find the information on how to compile and run JCoBox programs and the required jar files at `http://softech.informatik.uni-kl.de/Homepage/JCoBox`.

a) Write a server that handles multiple clients and accepts new clients at any time. All messages send by any client shall immediately be displayed at all clients. Prefix messages with a unique id for each client. Implement the `.bye`-command to close the connection between some client and the server.

- Do not use the suffix `.java` for your files, the jcobox compiler will overwrite them.
- In order to handle the network connections, do not use `ServerSocket` or Streams directly, use the wrapper classes provided on the webpage. You are free to extend the provided wrappers if you need more methods, but follow the advices in the comments.

b) Implement the `.history n` command.

Consider the following questions. If needed change your implementation such that these questions are negated.

- Does sending a long history influence the chat for other clients? Can it block the server or cause lags?
- Is it possible that the client, which requested the history, misses messages by others while receiving the history?

c) Which guarantees does your implementation provide about the order of messages? Compare with your thread-based solution of exercise 10.2.

Exercise 3 ThermoControl System Using Synchronous Messages

a) Implement the ThermoControl system from chapter 6.2 using synchronous messages. You can use the code provided on the webpage as a staring point.

   The class `AirConditioner` includes a thread for simulation of the system. Leave this part of the implementation unchanged and adapt the rest of the class. This way you can try your system after you have implemented the other components as well.

   - The `ControlPanel` should be implemented in Swing. It should show the current and the desired temperatur. Additionally the GUI should allow to change the desired temperatur using an increment and a decrement button.
   - The communication between the components should be realized using the class `SynchronousQueue` from the package `java.util.concurrent`.
   - The class `ThermoSensor` is complete and can be used as is.