Prof. Dr. A. Poetzsch-Heffter
Dipl.-Inform. Kathrin Geilmann

University of Kaiserslautern
Department of Computer Science
Software Technology Group

Advanced Aspects of Object-Oriented Programming (SS 2010)

Practice Sheet 11

Date of Issue: 24.06.10
Deadline: 1.07.10
(until 10 a.m. as PDF via E-Mail)

Exercise 1 Asynchronous vs synchronous Calls

```
class A {                    class B {                  class C {
   C c;                         C c;                       void m(int i) {
     void m1() {                 void m2() {                  System.out.println("i=" + i);
          c.m(5);                     c.m(9);              }
          b.m2();                 }                     }
     }                         }
 }
```

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();
b.c = c;
a.m();
```

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 RoboWorld

In a RoboWorld, robots drive around and try to collect gold nuggets.

The system consists of several CoBoxes. The CoBox `RoboWorld` handles all objects that are part of the world, the robots, the gold and the walls. It continuously simulates the movements of the robots and produces new gold nuggets. Whenever a nugget is produced, the world object sends a message to its registered observers. The simulated robots send out messages to their observer, whenever they hit or sense something. The CoBox `Main` sets up the world and starts the simulation. The CoBox `RoboWorldGUI` bundles all operations, objects and tasks that are used to draw the GUI.

To control the robots additional CoBoxes are used. A simple CoBox for controlling a robot is implemented by `SimpleController`. In the following you are allowed to change the `Main`-Cobox to modify the world setup and to write your own Controller - CoBoxes.

More information about JCoBox can be found at `http://softech.informatik.uni-kl.de/Homepage/JCoBox`.

Hint for all implementations: think about what can be run in parallel an, where to use cooperative multitasking and where to use CoBoxes and which kind of communication is needed?

a) Download the Roboworld implementation from the lecture's website. The implementation includes the JCoBox compiler and runtime and the sources for a robot world. Compile and run the World by using the provided scripts (compile.sh, run.sh).

b) Look at the provided example controller `SimpleController`. It avoids collisions of the controlled robot with the wall by changing the direction whenever a wall is sensed. Nevertheless, after a while the controlled robot collides with the wall, why?

c) Implement a cobox, that acts as a controller for a robot, i.e. it registers as observer for robot events and sends messages to one robot. Try to collect as much gold as possible. Test your implementation on the different example worlds, that you can create with the methods in the class `ExampleWorld`.

d) Implement a robot swarm, i.e. implement one or several controllers in one or more CoBoxes that work together to collect the gold.

Exercise 3 Distributed Programming with RMI

a) Have a look at the Sun RMI tutorial at `http://java.sun.com/docs/books/tutorial/rmi/index.html`

b) Can RMI programs handle callbacks? What is the advantage of RMI over Web Services?

c) Can you receive a reference to a remote object, that is not registered at a naming service? If so, how?