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 12 (Hints and Comments)

Exercise 1 Distributed Programming with RMI

a) -

b) In general callbacks are supported, but as RMI copies non-remote objects to the remote side, it is difficult to see whether a callback will occur or not. Direct callbacks can only occur if the first object is also a remote object, but indirect callbacks can occur if the controlflow passes via another remoteobject back to the first object.

According to the W3C: *A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP-messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards.*

RMI is directly supported by the language and uses real java-objects as parameters and message receivers. With some exceptions they behave like standard objects, whereas a webservice uses its own message format and it transfers messages not objects. Webservices cannot provide callbacks, because they don't have access to the object system.

c) As remote objects are not copied during transfer, the usage of a remote object as parameter or return value exposes the remote object, even if it has not been registered before. The name service only serves as a central repository to find objects by their name, it has to be used to establish the first connection between two machines. After that, references to remote objects may be passed freely around.

Exercise 2 RMI-Chatsystem

See provided sources.

Exercise 3 Swing

a) See provided sources.

b) The execution of a `SwingWorker` involves three threads:

- Current thread: The `execute()` method is called on this thread.
- Worker thread: The `doInBackgroud()` method is called on this thread.
- Event Dispatch Thread: All Swing related activities occur on this thread. Specifically, the methods `process()` and `done()` are invoked on this thread.

Typically, the interface between the worker and the GUI is very broad: The `process()` method, which is part of the worker and updates the state of the GUI, will access most of the elements of the GUI directly. Great care has to be taken that this access is thread-safe. Thread-safety can be accomblished by relocating all access to the GUI elements to the Event Dispatch Thread using `SwingUtilities.invokeLater` or `SwingUtilities.invokeAndWait` or the `process` and `done` methods of `SwingWorkerS`.