Übungen RMI

1) Schau dir die Folien “RMI_tutorial“ und das darin enthaltenen Codebeispiele genauer an und versuche zu verstehen wie RMI funktioniert. Zeichne eine Grafik, die alle wichtigen Schritte beschreibt und erkläre mit deinen eigenen Worten die Funktionsweise. Welche Rolle spielen dabei stub und skeleton?
   Informiere dich anschließen im Internet was die VM-Parameter codebase, hostname, useCodebaseOnly und security.policy bewirken und bei welchen Szenario sie benötigt werden. Welche Vor- und Nachteile hat das verwenden eines SecurityManagers?
   Programmiere die Beispiele dann nach und versuche die Programme auszuführen. Wie müssen RMI Anwendung ausgeführt werden?

2) a) Programmiere eine RMI Anwendung, die ein entferntes Objekt mit einer Methode quadrat(long) erzeugt. Die Methode wird mit einer Zahl als Parameter aufgerufen wird und berechnet das Quadrat dieser Zahl.
   Folgende Vorgehensweise wird empfohlen:
   1. Das Quadrat Interface (Quadrat.java) schreiben, welches das entfernte Objekt definiert.
   2. Die Klasse QuadratImpl (QuadratImpl.java) schreiben welche das Interface Quadrat implementiert. Sie soll dabei eine main()-Funktion beinhalten, die den RMISecurityManager erzeugt und installiert, falls er nicht schon installiert ist, und das entfernte Objekt bei der rmiregistry registiert.
   3. Das Klientenprogramm QuadratClient (QuadratClient.java) schreiben, welches zwei Argumente auf der Kommandozeile entgegennimmt: den Namen der Servermaschine auf welcher das entfernte Objekt registriert ist und die Zahl welche quadriert werden soll.

   b) Der Server soll eine weitere Methode enthalten, die generische Tasks lösen kann. Es soll ein Interface Task geben, das als Argument dem Server übergeben wird. Versuche anhand dieser Methode das Quadrat wie in Aufgabe 2a zu berechnen.

   Beantworte noch folgende Frage:
   a) Was beinhaltet das Interface java.rmi.Remote? Wieso?
   b) Welchen Unterschied gibt es zwischen Aufgabe 2a und Aufgabe2b und welchen Vor- und Nachteil haben diese unterschiedlichen Vorgangsweisen?
   c) Überprüfe ob ein Client mehrere Aufrufe simultan durchführen kann. Überprüfe auch ob dein Server auch mehrere Clients gleichzeitig abarbeiten kann.

3) Erstelle eine weitere RMI Anwendung mit den folgenden Anforderungen:
   a) Erstelle einen Server der eine Methode enthält, die auf eine Frage (als String übergeben) die Antwort „Die Antwort auf deine <Frage> ist wahrscheinlich 61“ zurückgibt.
   Diese Operation sollte sehr lange dauern und deshalb soll der Client nicht auf den Server warten. Implementiere eine asynchrone Methode, die zusätzlich zum String eine Objektreferenz (von einer Klasse die Remote implementiert) als Parameter übergeben bekommt, bei der die Callback-Methode aufgerufen wird. Die Objektreferenz sollte allerdings nicht exportiert werden. Nachdem der Client die Antwort erhalt sollte er beendet werden. Beschreibe auch den Callback-Mechanismus genauer!
   b) Schreibe einen Klienten der den Server verwenden kann

4) **Zusatzaufgabe:** Erstelle eine RMI-Anwendung die es zwei Angestellten einer Firma ermöglich von zwei unterschiedlichen Rechnen (einer in Amerika, einer in Europa) auf eine Datenbank zuzugreifen. Der Server sollte alle erlaubten Methoden für Änderungen in der Datenbank zur Verfügung stellen. Verwende dabei eine beliebige Datenbank, die du über den Java Database Connectivity Treiber (JDBC) ansprichst.

Abgabe:

Alle Aufgaben müssen selbständig gelöst und kommentiert werden. Bei identischen Aufgaben bekommen die betreffenden Schüler eine stark negative Note.
Abzugeben sind pro Aufgabe: Test der Funktionsweise (z.B. Screenshot, Kommandozeilenausgabe), Quellcode gut dokumentiert (u.a. wie das Programm kompiliert und ausgeführt werden kann), Executable Java File (für Java Projekte) für alle Clients und Server. Der Test muss in einer Datei namens readMe abgespeichert werden. Bei den Aufgaben wo auch Fragen zu beantworten sind, sollten auch diese in der readMe-Datei mit den entsprechenden Antworten vorhanden sein.
Eine weitere readMe-Datei aus der klar hervorgeht welche Aufgaben gelöst und warum evtl. Aufgaben nicht gelöst wurden (1 Absatz pro Aufgabe) muss vorhanden sein. In der readMe-Datei soll sich jeder Schüler auch selbst bewerten (Note von 1-10 geben).
Der Schüler sollte endsprechend seiner Note die Programme bei einer Prüfung erklären können. Eine Aufgabe zählt nur dann als abgegeben, wenn alle Dateien dieser Aufgabe vorhanden sind.

Abgabetermin: **02.04.2017 um 24 Uhr** mit Git. Das Repository muss RMI genannt werden. Für jede Übung muss ein Unterordner mit den entsprechenden Files erstellt werden. Dieser Ordner muss so genannt werden, dass eindeutig hervorgeht um welche Aufgabe es sich handelt (z.B. 01_HelloWorld für Aufgabe 1). Die readMe-Datei aus der hervorgeht welche Aufgaben gelöst wurden und der vorgeschlagenen Note, muss in der Wurzel-URL liegen. In den Unterordnern soll die readMe-Datei der entsprechenden Aufgaben vorhanden sein. Alle readMe-Dateien müssen als **PDF** abgegeben werden.