Übung 07: Client/Server mit RMI

Abgabetermin: 23. 5. 2022, 8:15

Aufgabe	Punkte	Abzugeben über Moodle
Übung 07	24	Java-Programm in Zip-Datei

Übung 07: Client/Server mit RMI (24 Points)

Nach *JayUnit* hat nun auch Ihre Kalenderanwendung viel Beachtung gefunden. Andere Tutoren verwenden sie mittlerweile ebenfalls zur Planung und haben auch eigenen Code zur Implementierung beigetragen. Die Institute und deren LVA-Leiter wittern aber jetzt auch eine Chance, die Koordination von und mit Ihren Tutoren zu vereinfachen. Künftig sollen alle Tutoren einen gemeinsamen Kalender verwenden, damit LVA-Leiter jederzeit über den Zeitplan der Übungskorrekturen informiert sind. Natürlich wird Ihnen die verantwortungsvolle Aufgabe der Implementierung dieses *SharedCalendar* übertragen.

Sie erhalten eine Implementierung der Kalenderanwendung aus Aufgabe 2. Erweitern Sie diese Implementierung sodass:

- das Datenmodell (*calendar.backend.CalendarModelImpl*) auf einem Server verwaltet wird, und
- Clients (*calendar.frontend.**), auch mehrere parallel laufende, über RMI auf das Datenmodell am Server zugreifen können.

Sie dürfen, und müssen zur Erfüllung der Aufgabe, jeden Teil des zur Verfügung gestellten Codes verändern. Beachten Sie etwa, dass das zur Verfügung gestellte Datenmodell zwar prinzipiell schon zum Zugriff von mehreren Threads bereit ist, aber keine eigenen Threads zum Aufrufen der *CalendarListener* angelegt werden. Außerdem wird momentan bei Beendigung der GUI auch die Verbindung des Datenmodells zur Datenbank terminiert.

Konkret sind folgende Schritte nötig:

- Erweitern Sie die Interfaces und Implementierung des Datenmodells zur Verwendung über RMI.
- Erweitern Sie die Multi-Threading Unterstützung des Datenmodells, sodass durch das Ausführen von Listeners keine Deadlocks entstehen können.
- Erstellen Sie einen Server, der das Modell-Objekt (*CalendarModel*) via RMI zur Verfügung stellt.
- Adaptieren Sie die GUI (*calendar.frontend.CalendarController*), sodass RMI-basierte Fehler (*RemoteException*) gefangen und dem User angezeigt werden. Sie können hierzu bestehenden Code zur Fehlerbehandlung wiederverwenden und erweitern.
- Adaptieren Sie die GUI (*calendar.frontend.CalendarController*), sodass diese beim Beenden eines Clients nicht das Datenmodell am Server schließt und dadurch den erfolgreichen Anschluss anderer Clients verhindert.
- Erweitern Sie die Client-Applikation (*calendar.frontend.CalendarApplication*), sodass diese auf das Datenmodell am Server zugreift und darauf basierend die GUI startet.

Hinweise:

- Eine Implementierung der Kalenderanwendung wird Ihnen zur Verfügung gestellt. Diese Implementierung enthält eine JavaFX-basierte GUI und eine Derby-basierte Datenbankanbindung.

Verwenden Sie diese Implementierung als Grundlage für die Implementierung dieser Hausübung. Das gleiche Framework wird auch bei Übungen 8 und 9 verwendet werden.

- Das zur Verfügung gestellte Zip-Archiv enthält neben dem Source Code der Kalenderanwendung auch eine *Maven* Konfigurationsdatei. Gängige Entwicklungsumgebungen wie Eclipse oder IntelliJ, die Maven-Unterstützung bieten, sollten aus dieser Konfigurationsdatei die benötigten Frameworks (JavaFX, Derby) auslesen und die notwendigen Dateien und Projekteinstellungen automatisch herunterladen bzw. setzen können. Z.B. in IntelliJ genügt: „Open“ -> <extrahiertes Zip-Archiv auswählen> um den Importvorgang anzustoßen. Sie dürfen die Abhängigkeiten aber bei Bedarf bzw. Problemen auch manuell konfigurieren. Beachten Sie bitte, dass das Framework zumindest Java 11 als Basis erwartet. Der Code sollte auch mit Java 8 funktionieren, allerdings wäre in diesem Fall wohl eine manuelle Korrektur der Abhängigkeiten notwendig, um zu Java 8 kompatible Versionen zu verwenden.
- Sie müssen sich nicht darum kümmern, das Modell-Objekt am Server freizugeben und den Server zu terminieren. Der Server kann endlos laufen, bis er manuell terminiert wird.
- Bei Beendigung der GUI soll die Verbindung zum Server terminiert und somit der Prozess des Client beendet wird. Achten Sie dazu darauf, dass etwaige Remote-Objekte wie Listeners in der GUI den Prozess nicht unnötig am Leben erhalten.
- Beachten Sie die in den Folien angemerkten Best-Practices zur Verwendung von RMI, wie etwa die Verwendung von Ids anstatt Objekten in Interfaces oder bei Objekt-Vergleichen.
- Zur Unterstützung zum Verständnis der Interaktion zwischen Datenmodell und GUI wird Ihnen die Klasse *DebugCalendarModel* zur Verfügung gestellt. Diese implementiert das *CalendarModel* Interface, leitet aber alle Methodenaufrufe an ein anderes Objekt weiter, welches ebenfalls dieses Interface implementiert, und gibt dabei die Parameter des Aufrufes sowie den aufrufenden Thread auf die Konsole aus.
- Achten Sie darauf, dass durch die Interaktion von Client und Server keine Deadlocks entstehen.
- Achten Sie auf eine saubere Implementierung.