Übung 7: Akka Bank

In dieser Übung soll die *Bank* aus den vorangehenden Übungen mit Hilfe von Akka realisiert werden. Wenn Sie so vorgehen wie beim Chat-Beispiel aus dem Unterricht, dann muss im Driver auch ein Aktor gestartet werden, welcher die Antworten vom Server entgegennehmen kann, und die Resultate können dann wie bei der WebSocket-Bank z.B. mit einer Queue ausgetauscht werden.

Anstelle eines lokalen Aktors kann aber auch das `ask`-Pattern verwendet werden, dann wird implizit ein lokaler Aktor gestartet. Das Resultat von `ask` ist ein (Scala)-Future, auf dem dann gewartet werden kann, bis die erwartete Antwort verfügbar ist:

```
Timeout  timeout = new Timeout(5, TimeUnit.SECONDS);
Future<Object> res = Patterns.ask(bankActor, message, timeout);
Object result = Await.result(res, Duration.Inf());
```

Beide Ansätze können verwendet werden, um die auf dem Command-Pattern aufgebaute Bank zu implementieren.

Auf Klientenseite muss jedoch zwingend ein Aktor gestartet werden, damit das `BankDriver2`-Interface implementieren werden kann (jenes, welches asynchrone Notifikationen vom Server unterstützt). Der Klient-Aktor wird die Notifikations-Meldungen entgegennehmen.

Auf dem Server existiert also ein Aktor, der alle Anfragen entgegennimmt und an die lokale Bank-Implementierung weiterleitet.

Interessanter ist es jedoch, wenn auf dem Server nicht nur *ein* Aktor aktiv ist, sondern wenn *jedes Konto* als eigener Aktor realisiert ist! Beim Aufruf von `getAccount` erhält man dann vom Server eine Remote-Referenz auf den entsprechenden Konto-Aktor. Dieser Ansatz würde es ermöglichen, die einzelnen Kontis auf verschiedene Rechner zu verteilen! Thread-Sicherheit ist dabei kein Problem mehr, denn die Operationen in einem Aktor (wie `deposit`, `withdraw`, `closeAccount` etc.) werden streng sequenziell ausgeführt. Trickreich ist dann jedoch die Methode `transfer`, da dabei zwei unterschiedliche Aktoren involviert sind. Wenn Sie dies lösen können, dann haben Sie viel über verteilte Systeme gelernt.

Abgabe: 03./05. Juni 2024
Besprechung: 10./12. Juni 2024