Professionelle Softwareentwicklung

Woche 6

Projekt

IMPORTANT	Die Abgabe erfolgt über AUAS (https://auas.cs.uni-duesseldorf.de) Abgabefrist: 08.06.2018 um 13:00 Uhr (Ortszeit Düsseldorf)

Aufgabe

Sie sollen eine Anwendung entwickeln und müssen sich dabei an die Regeln aus dem beigefügten Text *Object Calisthenics* von Jeff Bay halten. Die Regeln müssen dem Sinn nach befolgt werden, d.h., **wir werden keine rein "technisch korrekten" Lösungen, wie in der Vorlesung gezeigt wurde, akzeptieren**.

Die Regeln von Jeff Bay werden um eine weitere Regel ergänzt:

IMPORTANT	Regel 10: In der gesamten Codebasis existiert höchstens eine statische Methode.

Das Programm

Die Aufgabe ist die Entwicklung einer Software, mit der Gruppen eine gemeinsame Ausgabenkasse führen können, z.B. für eine Wohngemeinschaft, Reisegruppe, etc.

Es gibt eine Reihe von Regeln, die beachtet werden müssen. Wir betrachten das am Fall einer Städtereise von Simon, Thomas, Jens, Gerd, Kathrin, Christoph, Jessica und Sina sowie der Wohngemeinschaft von Christoph, Thomas und Markus.

- Als Projekt bezeichnen wir eine Gruppe von Personen, die unter sich Gelder aufteilen. Eine Person kann in mehreren Projekten sein, Christoph und Thomas sind zum Beispiel sowohl bei der Städtereise dabei, als auch Mitbewohner in der WG.
- Es können Kosten für die gesamte Gruppe, aber auch für Teilgruppen entstehen, z.B. nehmen alle Teilnehmer der Städtereise an der Stadtrundfahrt teil, aber Simon, Thomas und Jens haben keine Lust auf den Musicalbesuch.
- An einer Bezahlung können eine oder mehrere Personen beteiligt sein, zum Beispiel bezahlt Christoph die Miete für die WG. Die Musicalkarten auf der Städtereise waren aber recht teuer und Sina und Kathrin haben gemeinsam das Geld ausgelegt. Sina hat 500 Euro bezahlt, Kathrin hat den Rest von 364,50 Euro bezahlt.
- Jede Bezahlung soll in dem Programm nachvollziehbar sein. Dazu gehört, dass neben dem Betrag, dem Projekt und der Personengruppe unter der der Betrag aufgeteilt wird, auch der Grund und das Datum festgehalten werden. Als Beispiel hat Jens für die Stadtrundfahrt für alle am 30.10.2017 einen Betrag von 148 Euro bezahlt.
- Bei der Abrechnung wird ein Report ausgegeben, dem entnommen werden kann, wer wem wieviel Geld schuldet. Für jede Person wird eine Liste ausgegeben, anhand derer nachvollzogen werden kann, wie der Betrag zusammenkommt. Das genaue Format können Sie selber festlegen.

- Die Anzahl der Überweisungen soll klein gehalten werden, es darf auf keinen Fall mehr als eine Überweisung zwischen zwei Personen geben.
  Wenn Sie Zeit und Lust haben können Sie versuchen, ob Sie eine Optimierung finden können. Wenn zum Beispiel Jens Christoph 50 Euro schuldet und Christoph schuldet Thomas 50 Euro, könnte man statt zwei Überweisungen auch Jens das Geld direkt an Thomas überweisen lassen.
- Die Abrechnung für ein Projekt kann zu einem beliebigen Zeitpunkt erfolgen, danach gilt das Projekt entweder als beendet (Städtereise) oder es werden alle Konten auf 0 gesetzt (WG).

Bestandteile der Abgabe

Die Java Anwendung

Die Anwendung muss unter Verwendung des Application Plugin mit Gradle gebaut und mit `gradle run` gestartet werden. Der Einfachheit halber brauchen Sie keine Eingabe in Textform einzulesen. Stattdessen sollen Sie in **einer einzigen Methode** folgendes Szenario implementieren:

1. Simon, Thomas, Jens, Gerd, Kathrin, Christoph, Jessica und Sina nehmen an der Städtereise nach Hamburg teil. Das Projekt erfordert eine einmalige Abrechnung.
2. Markus, Thomas und Christoph wohnen in einer WG, die Abrechnung findet jeden Monat statt.
3. Christoph bezahlt die Miete in Höhe von 1275,80 Euro.
4. Markus zahlt 29,99 Euro für die Internet Flat
5. Thomas besorgt Bier für den Monat, er bezahlt 63,80 Euro. Markus trinkt kein Bier.
6. Thomas füllt auch den Kühlschrank für alle, er bezahlt 104,88 Euro.
7. Jens bezahlt bei der Städtereise am 30.10.2017 den Betrag von 148 Euro für die Stadtrundfahrt, an der alle teilnehmen.
8. Die Tickets für das Musical kosten 864.50 Euro, Sina bezahlt davon 500 Euro, der Rest wird von Kathrin übernommen. Es nehmen alle, ausser Simon, Thomas und Jens teil.
9. Thomas bezahlt 66 Euro für das Bier am Abend, Simon hatte Kopfschmerzen und ist im Hotel geblieben.
10. Die Reisegruppe kommt wieder zu Hause an. Gerd hat für Benzin 61,38 Euro ausgegeben, bei ihm sind Jens, Kathrin, Sina und Christoph mitgefahren. Simon hat den Rest der Gruppe mitgenommen und hat 54,43 Euro für Benzin ausgegeben.
11. Die Abrechnung für die Städtereise wird durchgeführt, damit ist das Projekt beendet.
12. Die erste Abrechnung für die WG wird durchgeführt.
13. Christoph bezahlt die Miete in Höhe von 1275,80 Euro.
14. Markus zahlt 29,99 Euro für die Internet Flat.
15. Die zweite Abrechnung für die WG wird durchgeführt.

Diese eine Methode braucht nicht die Regeln der Object Calisthenics beachten, darf aber ausschließlich eine Nutzereingabe simulieren, d.h. sie darf außer dem Erzeugen der Objekte und dem Aufruf der Methode zur Abrechnung keine Logik enthalten. Ihr Programm muss mit

beliebigen anderen Szenarien funktionieren.

Eine Projektdokumentation

Eine Beschreibung Ihrer Implementierung, in der Sie auf die folgenden Fragen eingehen:

- In welcher Methode ist das vorgegebene Szenario implementiert?
- Welche Klassen existieren? Welche Aufgaben haben diese Klassen? Wie spielen die Klassen zusammen?
- Wo gab es Probleme mit den Regeln der Object Calisthenics und wie haben Sie diese gelöst?
- Haben Sie die Object Calisthenics irgendwo verletzt? Wo und warum? Wenn Sie einen **guten** Grund für eine Verletzung vorbringen können, kann das ggf. akzeptiert werden.
- Welche anderen Designs haben Sie entwickelt, warum haben Sie sich für die abgegebene Fassung entschieden. Wenn Sie keine alternativen Entwürfe in Betracht gezogen haben, haben Sie etwas falsch gemacht!

Formalia

Die Abgabe muss eine Zip-Datei sein, es müssen folgende Bestandteile vorhanden sein:

1. Eine Datei im AsciiDoc-Format mit der Beschreibung Ihrer Implementierung. Wir erwarten von Ihnen, dass die Beschreibung weitestgehend frei von grammatikalischen und orthographischen Fehlern ist. Sie dürfen die Beschreibung in deutscher oder englischer Sprache anfertigen.
2. Der Java Code in den richtigen Unterverzeichnissen.
3. Die build.gradle Datei

Ihre Abgabe darf auf **keinen Fall** generierte Dateien beinhalten, z.B. jar-Dateien oder .class-Files. Die dürfen auch nicht Quelltexte mitliefern, die aus den notwendige(n) Bibliothek(en) stammen.

Video

Schauen Sie sich as Video des Parrot Refactorings (https://youtu.be/DwfvWO_c3tA) an.