SmsHandy-Projekt – Grundlagen der Programmierung 2 SoSe 2019

Das Projekt wird als Teil der Prüfungsvorleistung in den nächsten Wochen schrittweise durch zwei Studierende gemeinsam bearbeitet.

Die Abgabe des fertigen Projekts erfolgt über den Projektserver bis zum 03.06.2019 um 09:00 Uhr. Code von anderen Gruppen zu übernehmen gilt als Plagiat und wird als Betrugsversuch mit der Nichtanerkennung der Prüfungsvorleistung bewertet.

Die Klassen sollen sauber programmiert sein, d.h. mit aussagefähigen Variablennamen, korrekter Einrückung und vollständigen JavaDoc-Kommentaren. Insgesamt soll das Projekt nach dem Model-View-Controller-Muster (MVC) aufgebaut sein, wie es in der Lehrveranstaltung besprochen wurde. Zur Umsetzung der grafischen Oberfläche dient JavaFX.

Folgende Rahmenbedingungen sind dabei zwingend einzuhalten:

- **Die Abgabe wird ausschließlich über das entsprechende Repository auf dem Projektserver bis zum angegebenen Abgabedatum akzeptiert. Machen Sie die finale Version sowie die Versionen für die einzelnen Meilensteine (s.u.) mit einem entsprechenden Commit-Kommentar kenntlich. Ansonsten wird die letzte vor Ablauf der jeweiligen Frist hochgeladene Version zur Bewertung herangezogen.**
- **Aufgrund regelmäßiger Commits auf dem Projektserver muss nachvollziehbar sein, dass beide Studierende hinreichend an der Entwicklung beteiligt waren. Beispielsweise wird ein großer „Gesamt-Commit“ des vollständigen Projekts am Ende der Projektbearbeitungszeit durch einen der Studierenden nicht akzeptiert.**
- **Beide Studierende müssen Quellcode auf allen Ebenen des MVC-Musters entwickeln, was anhand der Commits nachvollziehbar sein muss.**

Meilensteine

Zu den nachfolgend genannten Meilensteinen muss jeweils der zugehörige Funktionsumfang in der Lehrveranstaltung vorgeführt werden können. Ist das zu diesem Datum nicht möglich, gilt das Projekt als gescheitert und dieser Teil der Prüfungsvorleistung als nicht bestanden. Es darf natürlich gerne schon vorausgearbeitet werden und mehr umgesetzt werden, als der aktuelle Meilenstein verlangt. Allerdings muss das Projekt bei den Meilensteinkontrollen lauffähig sein. Die Inhalte und Abgabe-/Kontrolltermine der einzelnen Meilensteine können Sie der folgenden Tabelle entnehmen:

Meilenstein	Inhalt (Projektaufgaben)	Datum
1	1 und 2	13.05.2019
2	3 und 4	20.05.2019
3	5	27.05.2019
4	6	17.06.2019

Tabelle 1: Übersicht über Projektmeilensteine

Projektaufgaben

1. Grundlegendes fachliches Klassenmodell (Meilenstein 1)

Entwickeln Sie mit Eclipse ein neues Projekt mit dem Namen `SmsHandy_Nachname1_Nachname2` (wobei `Nachname1` und `Nachname2` durch die Nachnamen der beiden beteiligten Studierenden zu ersetzen sind), in dem ein Handynetz simuliert wird, dessen Handys ausschließlich SMS senden können.

Das Projekt enthält zunächst die fünf fachlichen Modellklassen:

- `SmsHandy`
- `PrepaidSmsHandy`
- `TariffPlanSmsHandy`
- `Provider`
- `Message`.

Alle Klassen befinden sich in dem Package `smsHandy`.

Legen Sie die Klassen einschließlich Dokumentation selbst an. Als Ausgangspunkt bekommen Sie dazu sowohl die Klassendokumentation als Javadoc, die alle öffentlichen Elemente der Klassen beschreibt, in der ZIP-Datei als auch das Klassendiagramm als Teil dieser Anleitung. Sie werden beide Informationsquellen für eine korrekte Umsetzung benötigen. Wie in der Telekommunikation üblich sind alle Namen (Klassen, Variablen, Methoden) in Englisch. Bitte halten Sie sich in diesem Projekt an diese Vorgabe. Kommentare können aber auch auf Deutsch verfasst werden.

Beachten Sie dabei, dass die Klassen noch nicht für die Nutzung des Binding-Mechanismus in JavaFX angepasst wurden. Nehmen Sie dies selbstständig vor.

Nutzen Sie zur Verwaltung der Teilnehmer (Assoziation `subscriber`) in der `Provider`-Klasse eine geeignete `Map`, die die Rufnummer als Schlüssel verwendet.

2. Kostenverwaltung der SMS (Meilenstein 1)

Die Kosten des SMS-Versand werden unterschiedlich gehandhabt, je nachdem, ob es sich um ein Prepaid- oder ein Vertrags-Handy handelt.

Beim Prepaid-Handy wird das Guthaben jedes Teilnehmers beim `Provider` verwaltet. Dazu sollten Sie eine weitere `Map` mit dem Namen `credit` verwenden. Das Guthaben wird dabei durch `int`-Werte(!) verwaltet.

Beim Anlegen eines neuen `PrepaidSmsHandys` gibt es ein Startguthaben von 100 Einheiten, das mit jeder über den `Provider` versendeten SMS jeweils um 10 Einheiten vermindert wird. Die Konstruktorsignatur soll sein: `SmsHandy (String number, Provider provider)`. Führen Sie weiterhin eine Methode `deposit(int amount)` im `PrepaidSmsHandy` ein, die das Guthaben auflädt. Eine gleichnamige Methode soll es auch in der Klasse `Provider` geben. Diese hat die Signatur `void deposit(String number, int amount)`, wobei `number` die Kennung des Handys darstellt, dessen Guthaben geändert werden soll.

Modifizieren Sie außerdem die `send(..)`-Methode im `Provider` so, dass beim Senden einer SMS an die Nummer *101# eine `Message` mit dem aktuellen Guthaben zurückgesendet wird. Diese besondere SMS ist kostenfrei und wird auch nicht in der Liste gesendeter SMS abgelegt.

Mit der Methode sendSmsDirect(…) wird eine SMS direkt von einem Handy zum nächsten übermittelt, ohne über den Provider zu gehen. Damit entstehen dabei auch keine Kosten.

Das Vertrags-Handy (`TariffPlanHandy`) verfügt über eine feste Anzahl von Frei-SMS (100), die im Handy-Objekt verwaltet werden. Bei jedem Versand einer Nachricht wird eine Frei-SMS abgezogen. Dass diese nach einem bestimmten Zeitraum erneuert werden sollten, wird in dieser Version des Projekts noch nicht berücksichtigt.

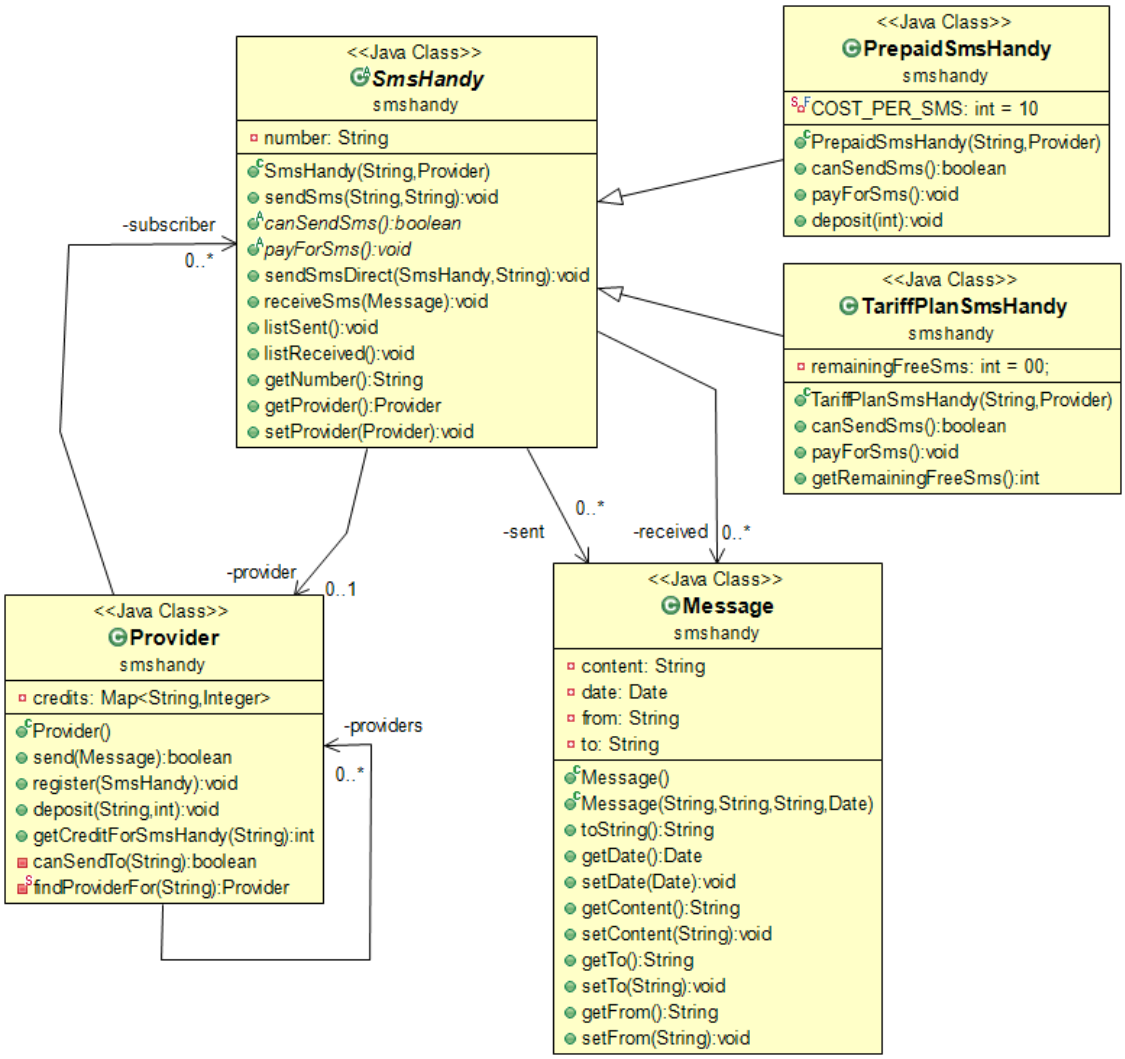

Abbildung 1: Klassendiagramm des SmsHandy-Projekts

3. Verwaltung der Provider (Meilenstein 2)

Verwalten Sie mehrere `Provider` in einer statischen `ArrayList<Provider>` in der Klasse `Provider` namens `providerList`. Jedes Mal, wenn ein neues `Provider`-Objekt angelegt wird, soll es sich im Konstruktor selbst in diese Liste eintragen. Wenn jetzt eine SMS an ein Handy gesendet werden soll, das bei einem anderen `Provider` registriert ist, dann soll diese SMS über diesen `Provider` weitergereicht und zugestellt werden. Modifizieren Sie dazu die Methode `send()` in `Provider` und führen Sie zwei neue Methoden in der Klasse `Provider` ein:

- `private boolean canSendTo(String receiver)` liefert genau dann `true` zurück, wenn ein Teilnehmer mit der Rufnummer `receiver` bei diesem `Provider` registriert ist.
- `private static Provider findProviderFor(String receiver)` liefert den `Provider` zurück, bei dem der Teilnehmer mit der Rufnummer `receiver` registriert ist, oder `null`, wenn es die Nummer nicht gibt. Gehen Sie dazu die statische Liste der `Provider` durch und testen Sie mit `canSendTo(…)`, ob der `Provider` den Teilnehmer registriert hat.

Wird ein `Provider` gelöscht, wird die Zuordnung bei den verbundenen `SmsHandys` entfernt. Solange diese nicht einem neuen `Provider` zugeordnet werden, soll für diese Handys das Senden von SMS nicht möglich sein.

4. Tests und Exceptions (Meilenstein 2)

Setzen Sie Exceptions durchgängig im Projekt an geeigneten Stellen ein.

Testen Sie die öffentlichen Methoden aller fachlichen Modellklassen (außer Getter und Setter) mit dem JUnit-Framework.

5. Grafische Nutzeroberfläche (Meilenstein 3)

Entwickeln Sie eine grafische Oberfläche mithilfe von JavaFX mit in FXML entwickelten Views und zugehörigen Controller-Klassen. Damit sollen die in den fachlichen Modellklassen entwickelten Funktionalitäten über die GUI nutzbar gemacht werden. Mindestens muss die GUI folgende Punkte ermöglichen:

- Auflisten aller vorhandenen Provider und SmsHandys einschließlich der Art des Handys
- Hinzufügen neuer Provider und SmsHandys
- Löschen von Providern und SmsHandys
- Anzeige des restlichen Guthabens bzw. der Frei-SMS der Handys
- Zuordnung von SmsHandys zu Providern ändern

6. Erweiterung der grafischen Nutzeroberfläche (Meilenstein 4)

Ergänzen Sie die GUI um die folgenden Punkte:

- Schreiben und Versenden von Nachrichten zwischen Handys (Der Versand der Nachricht zwischen den Handys soll über eine einfache Animation visualisiert werden.)
- Anzeigen der gesendeten und empfangenen Nachrichten eines Handys
- Aufladen des Guthabens eines SmsHandys