www.clean-coding-cosmos.de
Reik Oberrath, Jörg Vollmer
7. März 2016

CSV-Parser

Ein Clean-Coding-Dojo

Die Klassen `CSVParserClean` und `CSVParserDirty` beinhalten jeweils eine Methode `parseLine`, die dazu dient, eine Zeile einer CSV-Datei in ihre Tokens zu zerlegen. Beispiel:

```
Ich;verwende;das;“Ausführungszeichen “” und“;das;“Semikolon ; gern.“
```
—>
[Ich] [verwende] [das] [Anführungszeichen “ und] [das] [Semikolon ; gern.]

An den Sonderfällen ist zu erkennen, dass die Aufgabe nicht ganz trivial ist. Nun gibt es Fälle, die auch nach der WikiPedia-Definition[1] nicht ganz eindeutig sind. Beispiel:

```
Der;“Pe“ter;programmiert;gern.
```

Die o.g. Parser erzeugen beide die Token-Folge:

[Der] [Pe“ter] [programmiert] [gern.]

Aufgabe 1

Ein Anforderungsmanager entdeckt, dass Excel beim Einlesen der CSV-Datei daraus aber

[Der] [Peter] [programmiert] [gern.]

macht, und bittet die Entwickler, das entsprechend anzupassen.
In den exakt gleichen Testklassen `CSVParserCleanTest` und `CSVParserDirtyTest` befinden sich bereits zwei Tests (die letzten beiden), die diese Excel-Variante erwarten. Also lautet die Aufgabenstellung kurz:

1. „Mach, dass alle Tests grün sind!“
2. Der Parser soll sich exakt so verhalten wie Excel.[2]

---

[1] https://de.wikipedia.org/wiki/CSV_(Dateiformat)

[2] So lassen sich beliebig viele Testfälle auch selbst erzeugen; den erwarteten Wert liefert Excel.

Aufgabe 2

Die jeweiligen Klassen sind so zu erweitern, dass man nicht nur eine Zeile, sondern auch eine ganze CSV-Datei einlesen können soll. Die Schwierigkeit liegt wieder darin, dass Umbrüche, die normalerweise die Zeilen identifizieren, auch innenhalb von CSV-Elementen vorkommen dürfen, sofern sie sich (wie das Semikolon) innerhalb eines String befinden, der durch Anführungszeichen eingeschlossen ist.

Hinweise für den Aufgabensteller

Die beiden o.g. Klassen behalten einmal eine saubere Implementierung der Aufgabe und einmal eine schmutzige. Man könnte zwei Gruppen bilden. Jede einzelne Person hat ein bestimmte Zeit, die Aufgabe zu lösen. Wer fertig ist, kann ein E-Mail an den Veranstalter senden. Am Schluss werden die durchschnittlichen Zeiten gemessen. Dazu ist das Java-Projekt in zwei Teile zu zerlegen. Die eine Gruppe bekommt die eine Klasse `CSVParserClean` einschl. Testklasse, die andere die Klasse `CSVParserDirty`.

Varianten

Falls viele Teilnehmer vorhanden sind, könnte man noch eine weitere Gruppe einrichten, die die Klasse `CSVParserDirtyWithComments` erhält. Sie enthält den gleichen unsauberen Code wie `CSVParserDirty`, nur das dieser ausführlich kommentiert ist. Somit ließe sich auch der Sinn und Zweck von Kommentaren untersuchen.

Ein weitere zusätzliche Gruppe bekommt auch die Klasse `CSVParserClean`, aber noch eine zusätzliche PDF-Datei, die den Algorithmus ausführliche erklärt. Das wäre sozusagen eine Luxus-Variante von „clean“.

Technische Voraussetzungen

Programmiersprache: Java (ab Version 7)
Build-System: Maven (ab Version 2)