Aufgaben „Reguläre Ausdrücke“

Zielsetzung der Aufgaben

Diese Aufgaben schaffen eine Übungsgrundlage für den Umgang mit regulären Ausdrücken. Nach einer grundlegenden Einführung in die Thematik wird die Konstruktion von regulären Ausdrücken für verschiedene Zwecke sowie die Anwendung auf Zeichenketten aus unterschiedlichen Quellen geübt.

Gleichzeitig werden bei dieser Aufgabe nochmals der Umgang mit (Text-)Dateien und Collections, sowie die Handhabung des Properties-Objektes beübt. An "handwerklichen" Fertigkeiten ist insbesondere der Import von Fremdcode in das eigene Projekt gefordert.

Zielsetzung dieses Dokuments

Dieses Dokument beschreibt die Programmieraufgaben vollständig. Eventuell erforderliche Klassendiagramme sind eingebettet.

**Aufgabe "Autokennzeichen identifizieren"**

Autokennzeichen eines Staates folgen i.a. einem gemeinsamen Muster. Für eine begrenzte Auswahl an Staaten sind diese Muster unter Umständen so unterschiedlich, dass aus dem Muster auf den Staat geschlossen werden kann.

In dieser Aufgabe gilt es, die (fiktive) Protokolldatei einer TollCollect-[Mautbrücke](http://de.wikipedia.org/wiki/Mautbr%C3%BCcke) auszuwerten und die Nationalitäten der Kennzeichen den jeweiligen Staaten zuzuordnen.

Vorarbeiten

*Übungskünstlichkeit:* Die Kennzeichen in dieser Aufgabe entsprechen den unten angegebenen Mustern, können aber ungültig sein (z.B. führende Nullen in Zahlen, Buchstabengruppen, die nicht vergeben werden) oder zusätzliche Zeichen enthalten, die in realen Kennzeichen nicht enthalten sind. Dies ist der künstlichen Erzeugung geschuldet und soll nicht weiter beachtet werden.

Der beiliegende Code enthält u.a. Textdateien, die die jeweiligen Regions-Präfixe enthalten:



Zunächst sind geeignete Ausdrücke zu entwerfen:

1. [Deutsche Autokennzeichen](http://de.wikipedia.org/wiki/Kfz-Kennzeichen_(Deutschland)) bestehen aus
   * einem Regions-Präfix
   * einem Bindestrich
   * einer frei wählbaren Buchstabengruppe aus mindestens einem, maximal zwei Buchstaben
   * einem Leerzeichen
   * einer Zahl mit mindestens einer, höchstens jedoch vier Ziffern

Hier den regulären Ausdruck bauen

[A-Z][-][A-Z]{2}[ ][0-9]{4}

1. [Schweizer Autokennzeichen](http://de.wikipedia.org/wiki/Kontrollschild_(Schweiz)) bestehen aus
   * einem Regions-Präfix
   * einem Bindestrich
   * einer Zahl mit zwei bis drei Ziffern
   * einem Leerzeichen
   * einer Zahl mit genau drei Ziffern

[A-Z][-][0-9]{2,3}[ ][0-9]{3}

1. [Österreichische Autokennzeichen](http://de.wikipedia.org/wiki/Kfz-Kennzeichen_(%C3%96sterreich)) bestehen aus
   * einem Regions-Präfix
   * einer Buchstaben- und Zahlenkombination mit mindestens zwei, höchstens jedoch fünf Stellen

[A-Z][A-Z0-9]{2,5}

1. [Belgische Autokennzeichen](http://de.wikipedia.org/wiki/Kfz-Kennzeichen_(Belgien)) bestehen aus
   * einer Zahl zwischen 1 und 9
   * einem Bindestrich
   * einer dreistelligen Buchstabengruppe
   * einem Bindestrich
   * einer Zahl zwischen 1 und 7
   * einer zweistelligen Zifferngruppe

[1-9][-][A-Z]{3}[-][1-7][0-9]{2}

1. [Italienische Autokennzeichen](http://de.wikipedia.org/wiki/Kfz-Kennzeichen_(Italien)) bestehen aus
   * einer zweistelligen Buchstabengruppe
   * einem Bindestrich
   * einer dreistelligen Zahl
   * einem Bindestrich
   * einer zweistelligen Buchstabengruppe

[A-Z]{2}[-][0-9]{3}[-][A-Z]{2}

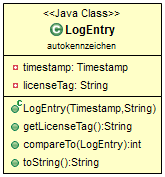
Die oben gefundenen Ausdrücke werden in einer Properties-Datei abgelegt, und zwar mit dem zweistelligen Ländercode nach [ISO-3166](http://de.wikipedia.org/wiki/ISO-3166-1-Kodierliste) als Schlüssel und dem regulären Ausdruck als Wert.

**Erzeugung und erste Analyse der Protokolldaten**

Als nächstes muss mit dem beiliegenden Code eine Protokolldatei erzeugt werden. Dazu dient die Klasse FileGenerator mit ihrer statischen Methode generateLogFile(fileName, lines). Das fertige Programm soll in jedem Durchlauf eine neue Protokolldatei erzeugen, die mindestens 10000 Zeilen lang sein soll.

Analysieren Sie eine Zeile typische Zeile der Protokolldatei, und überlegen Sie sich einen geeigneten regulären Ausdruck, um die Zeile in ihre aussagekräftigen Hauptbestandteile zu zerlegen. Welche Bestandteile sind dies? Welche Methode nutzen Sie zur Zerlegung?

Programmierung der Klasse für die Datenhaltung

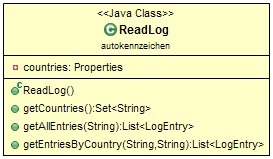
Log-Einträge bestehen aus einem Zeitstempel ([java.sql.Timestamp](http://docs.oracle.com/javase/7/docs/api/java/sql/Timestamp.html)) und einem String für das Kennzeichen. Sie sind untereinander vergleichbar ([Comparable](http://docs.oracle.com/javase/7/docs/api/java/lang/Comparable.html)), ihre Reihenfolge richtet sich dabei nach den Zeitstempeln.

Sie verfügen über einen Getter für das Kennzeichen und eine toString()-Methode, die einen String in der Form

2014-01-01 09:27:09.0 : DN-B 585

zurückgibt.

Programmierung der Klasse für das Einlesen der Logdatei

Die ReadLog-Klasse besitzt ein [Properties](http://docs.oracle.com/javase/7/docs/api/java/util/Properties.html)-Objekt, das bei der Konstruktion aus der o.a. Properties-Datei befüllt wird.

Es ist möglich, bei der ReadLog-Klasse das Set aus Strings abzufragen, das die Ländercodes entält.

Es ist möglich, eine Log-Datei unter Angabe eines Pfad-Strings komplett einzulesen, hierbei entsteht eine Liste aus Log-Einträgen.

Es ist möglich, nur die Einträge für ein bestimmtes Land aus einer bestimmten Datei zu bekommen. Dazu dient die Methode getEntriesByCountry(), die ein Länderkürzel und einen Pfad-String übernimmt. Falsche bzw. nicht vorhandene Länderkürzel werden mit einer IllegalArgumentException quittiert. Wie bei getAllEntries() entsteht eine Liste aus Log-Einträgen (allerdings nur solche, die zu dem Ländercode passen).

Die ReadLog-Klasse soll keinesfalls die gesamte Datei im Speicher halten!

Programmierung eines Starters

Der Starter enthält die main()-Methode. Hier wird wie oben angeführt, zunächst eine passende Log-Datei erzeugt.

In einem zweiten Schritt wird eine Map aus Listen(!) aus Log-Einträgen (Ländercode : Liste der Kenn­zeichen für dieses Land) angelegt und mit Hilfe der Methoden eines ReadLog-Objekts befüllt.

Am Ende soll etwa folgendes ausgegeben werden:

Reading for DE

Reading for AT

Reading for CH

Reading for IT

Reading for BE

Found DE : 2021

Found AT : 1967

Found CH : 2023

Found IT : 1961

Found BE : 2028

The File has entries 10000 and 10000 entries were matched.

Oben sind Statusmeldungen zu sehen (was im Moment gelesen wird, optional), dann folgt die Angabe, wie viele Kennzeichen für jedes Länderkürzel gefunden wurden. Die letzte Zeile vergleicht die Anzahl aller Zeilen der Logdatei mit der Summe der Anzahlen pro Land.

… und jetzt wissen Sie, warum

1. Die Identifikation des Herkunftslandes eines Autokennzeichens nicht über die Form des Kennzeichens, sondern über ein separates Merkmal (EU-Kennzeichen oder National­i­täts­plakette/-aufkleber) gelöst wird
2. Es für Datenanalysen Datenbanken gibt.