Aufgabe 2 IATACodes

Lernziele

1. Das Wissen über Zeichenketten und reguläre Ausdrücke aktiv anwenden.
2. Dateien mit einem Scanner auswerten können.
3. Vorgegebenen Quelltext anwenden können.
4. Ersten Umgang mit Collection Klassen üben.
5. Packages und Sichtbarkeiten anwenden.
6. Abstrakte Klassen und Interfaces in Java nutzen.

Hintergrund

Im Flugverkehr haben sich für Flughäfen und Fluglinien die sogenannten IATA-Codes etabliert. IATA steht für *International Air Transport Association*, ein Dachverband der Fluggesellschaften. IATA-Codes für Flughäfen sind 3-stellige Codes, die aus Großbuchstaben bestehen. IATA-Codes für Fluglinien sind 2-stellige Codes, die mit einer Ziffer beginnen, der ein Großbuchstabe folgt.

Quelle für IATA-Codes

Eine Quelle für IATA-Codes, in der die Codes als HTML-Ressourcen gespeichert sind, ist Wikipedia. Die Startseite für die Airport-Codes ist https://de.wikipedia.org/wiki/Liste_der_IATA-Flughafen-Codes, die für die Airline-Codes ist https://de.wikipedia.org/wiki/Liste_der_IATA-Airline-Codes.

Die IATA-Airline-Codes sind alle auf in einer HTML-Seite enthalten. Die Tabelle enthält u.a. den Code, die Airline und den Hauptsitz (das Land der Airline). Uns interessieren nur diese 3 Einträge.

Code	Fluggesellschaft	Land	Bemerkung
0D	Darwin Airline	Schweiz	neuer IATA-Code ist F7
0J	JetClub	Schweiz	Betrieb 2011 eingestellt
1A	Amadeus Global Travel Distribution	Spanien	
1B	Abacus International	Singapur	
1C	EDS Information Business	Schweiz	
1D	Radixx Solutions International	USA	
1E	Travelsky Technology	China	
1F	INFINI Travel Information	Japan	
1G	Galileo International (Galileo Host)	USA	
1I	Netjets	USA	
1I	Deutsche Rettungsflugwacht	Deutschland	

Anders verhält es sich für die IATA-Airport-Codes. Diese sind auf insgesamt 26 Seiten – eine Seite für jeden Anfangsbuchstaben der Codes – verteilt. Die Seite für die Codes beginnend mit 'A' hat die URL https://de.wikipedia.org/wiki/Liste_der_IATA-Codes/A. Alle anderen Seiten haben einen analogen URL-Aufbau (gleicher Präfix, Suffix gleich dem Anfangsbuchstaben der Codes).

Wir wollen aus der Tabelle nur die 1'te, 3'te und 6'te Spalte (den IATA-Code, den Namen des Flughafens und das Land) extrahieren.

IATA	ICAO	Flughafen	Ort	Region	Land
AAA	NTGA	Flughafen Anaa	Anaa	Tuamotu-Archipel	Französisch-Polynesien
AAB	YARY	Flugplatz Arrabury	Arrabury	Queensland	Australien
AAC	HEAR	Flughafen al-Arisch, *International Airport*	al-Arisch	Schimal Sina	Ägypten
AAD		Flughafen ad-Dabbah	ad-Dabbah	asch-Schamaliyya	Sudan
AAE	DABB	Flughafen Annaba, *Rabah Bitat Airport*, *Les Salines Airport*	Annaba (El Mellah)	Annaba	Algerien
AAF	KAAF	Flughafen Apalachicola, *Municipal Airport*	Apalachicola	Florida	Vereinigte Staaten
AAG	SSYA	Flughafen Arapoti	Arapoti	Paraná	Brasilien
AAH	EDKA	Flugplatz Aachen-Merzbrück	Würselen	Nordrhein-Westfalen	Deutschland
AAI	SWRA	Flughafen Arrais	Arraias	Tocantins	Brasilien
AAJ		Cayana Airstrip	Awaradam		Suriname
AAK	NGUK	Flughafen Aranuka	Aranuka	Gilbertinseln	Kiribati
AAL	EKYT	Flughafen Aalborg	Aalborg	Nordjylland	Dänemark
AAM	FAMD	Flughafen Mala Mala	Mala Mala	Mpumalanga	Südafrika

Tabellenzeilen, in denen der Name des Flughafens fehlt (wie z.B. AEU), sollen ignoriert werden.

AER	URSS	Flughafen Sotschi	Sotschi	Südrussland	Russland
AES	ENAL	Flughafen Vigra	Ålesund (Vigra)	Møre og Romsdal	Norwegen
AET	PFAL	Allakaket Airport	Allakaket	Alaska	Vereinigte Staaten
AEU			Abu Musa		Iran bzw. Vereinigte Arabische Emirate
AEX	KAEX	Alexandria International Airport	Alexandria	Louisiana	Vereinigte Staaten
AEY	BIAR	Akureyri Airport	Akureyri	Norðurland eystra	Island

Da für beide Code-Typen die Beschreibung in HTML vorliegt, müssen wir ein Programm schreiben, das mit Hilfe regulärer Ausdrücke, den Inhalt zwischen den HTML-Tags extrahiert.

Damit zur Entwicklungszeit der Traffic durch wiederholte Programmdurchläufe und –tests auf den Wikipedia Seiten nicht zu groß wird, sind alle HTML Quellen lokal in dem mitgelieferten Projekt enthalten.

Für die Airline-Codes: 1 Datei ***iata_airline.htm*** im Package ***iata.airline***

Für die Airport-Codes: 26 Dateien, 1 Datei für jeden Buchstaben des Alphabets. ***A.htm … Z.htm*** im Package ***iata.airport***

NUTZEN SIE NUR DIE LOKALEN DATEIEN!

Aufgabenstellung

In dem mitgelieferten Projekt finden Sie 3-Packages: ***iata***, ***iata.airline,*** und ***iata.airport.***

Package iata

- Die **abstrakte Klasse** ***AbstractIataCollectionReader*** spezifiziert die Schnittstelle für alle Objekte, die IATA-Objekte aus externen Quellen (Dateien oder Webseiten) lesen.

    - Die Klasse hat eine Erzeugermethode ***getInstance***, die je nach Parameterwert ein Objekt von Typ ***IataAirlineCollectionReader*** oder ***IataAirportCollectionReader*** zurückgibt. Das Entwurfsmuster, das hier angewendet wird, heißt auch **Factory-Pattern** (Erläuterung folgt in der Vorlesung über Klassen und Interfaces).

    - Die Klasse fordert von den Subklassen 2 read-Methoden zu implementieren.
        - ***readLocalCollection***: diese Methode soll aus allen Dateien, die die IATA-Codes enthalten, eine Collection von ***Iata***-Objekten erzeugen und diese Collection als Ergebnis zurückgeben. Die Methode hat ***public*** Sichtbarkeit.
        - ***readSingleCollection(URL url)***: diese Methode erzeugt aus einer Datei, die durch die ***url*** beschrieben wird, eine Collection von ***Iata***-Objekten und liefert die Collection als Ergebnis zurück. Dies ist sinnvoll für Reader, die die Sammlung aus den Inhalten vieler Einzeldateien erzeugen. (Wie aus einem Dateinamen eine URL wird steht in den Hilfestellungen). Die Methode hat ***protected*** Sichtbarkeit.

- Das **Interface** ***Iata*** ist der gemeinsame Supertyp für alle ***Iata***-Objekte (wir verwenden nur 2, es gibt aber z.B. auch IATA-Codes für Zuglinien). Alle ***Iata***-Objekte müssen Auskunft über den Code, den Namen und das Land geben können.

- Die Klasse ***IataCollectionUtility*** enthält eine Methode, um die Collections in eine Datei zu schreiben. Da die Anzahl der IATA-Codes z.T. sehr groß ist, sind die Dateien eine Hilfestellung bei der Fehlersuche.

- Die Klasse ***IataMain***

    - Erzeugt zunächst mit Hilfe der Erzeugermethode ***AbstractIataCollectionReader*** Objekte der konkreten Subklassen ***IataAirlineCollectionReader, IataAirportCollectionReader*** und
    - ruft dann die Methoden auf den Objekten der Subklassen auf. Dabei werden die Dateien ***iata_airport_list, iata_airport_list_S***, und ***iata_airline_list*** erzeugt.
    - Damit die Methode ***readSingleCollection*** auf einem ***IataAirportCollectionReader*** Objekt in dieser Klasse aufgerufen werden kann, müssen Sie die Sichtbarkeit die Methode in der Klasse ***IataAirportCollectionReader*** und **nur dort** auf **public** erweitern. Die Subklassen entscheiden darüber, ob die Außenwelt auf einzelne Quellen zugreifen darf. Wir verwenden hier denselben Trick, den auch ***Object*** in Java mit der

Methode ***clone*** anwendet. ***Clone*** ist **protected**, so dass die Subklassen entscheiden können, ob sie das Kopieren zulassen wollen.

- Die Klasse ***Testdaten***, die die unterschiedlichen Darstellungen für Airport-Name, Airport-Land und Airport-Code zeigt. Ihre Lösung muss alle diese Varianten in korrekte Darstellungen umwandeln.

Was ist zu tun?

Sie sollen die folgenden Klassen entwerfen, die korrekten Vererbungsbeziehung herstellen und die geforderten Methoden implementieren:

1. In den Packages ***iata.airline*** und ***iata.airport***
   - Je 1 Reader-Klasse in den (***IataAirlineCollectionReader***, und ***IataAirportCollectionReader***)
   - Je 1 Klasse, deren Objekte die IATA-Codes mit Zusatzinformation (siehe Beschreibung zu den Tabellen) speichern. (***IataAirline, IataAirport***)

2. Die Methoden der Klassen im Einzelnen:
   a. ***IataAirline*** und ***IataAirport***
      i. Methoden des Interfaces ***Iata***
      ii. Konstruktor, dem alle Eigenschaften des IATA-Codes übergeben werden
      iii. ***toString***

   b. ***IataAirlineCollectionReader***:
      i. ***readLocalCollection***: Die lokale Datei ***iata_airline.htm*** enthält die IATA-Airline Codes und liegt im gleichen Package wie die Klasse. Verarbeiten Sie die Datei mit einem Scanner, wie dieses in der Vorlesung über reguläre Ausdrücke gezeigt wurde. Das Ergebnis ist eine Liste von ***IataAirline***-Objekten.

      ii. ***readLocalCollection (URL)***: Der Parameter ist eine URL, die die lokale Datei ***iata_airline.htm*** beschreibt. Das Ergebnis der Methode ist eine Liste von ***IataAirline***-Objekten.

   c. ***IataAirportCollectionReader***:
      - ***readLocalCollection***: Die lokalen Dateien ***A.htm*** bis ***Z.htm*** enthalten die IATA-Airport Codes. Die Dateien liegen im gleichen Package wie die Klasse. Verarbeiten Sie alle Dateien und fügen Sie die entstehenden Teillisten in einer gemeinsamen Liste zusammen. Verarbeiten Sie die einzelnen Dateien mit einem Scanner. Das Ergebnis ist **eine** Liste von ***IataAirport***-Objekten.

      - ***readLocalCollection (URL)***: Der Parameter ist eine URL, die eine lokale Datei, z.B. ***U.htm***, beschreibt. Das Ergebnis der Methode ist eine Liste von ***IataPort***-Objekten, deren Code mit "U" beginnt. Das Ergebnis ist eine Liste von ***IataAirport***-Objekten. Zeilen, die keinen Namen für den Airport enthalten, sollen ignoriert werden.

Hilfestellungen:

1. Einlesen von lokalen Dateien und Umwandeln in eine URL (nur in Objektmethoden). Der Dateiname wird ohne Pfad angegeben (z.B., ***A.htm***):

   URL url = getClass().getResource(dateiname);

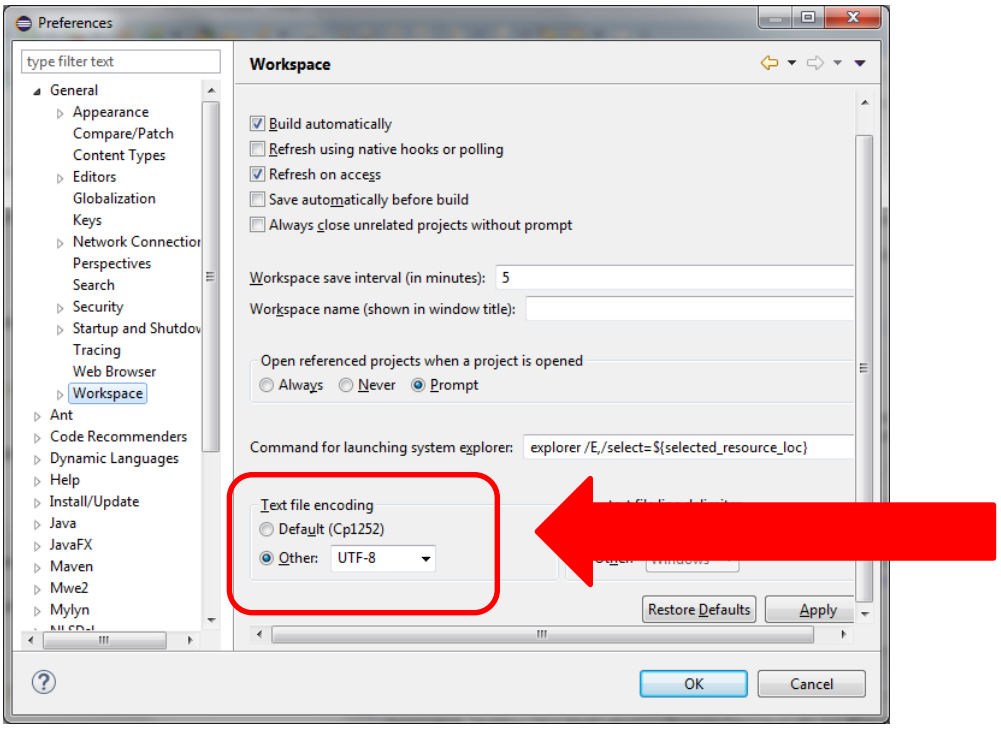
2. Verwenden einer URL als Quelle für einen Scanner mit Zeichensatz UTF-8 (für korrektes Lesen von Umlauten und Sonderzeichen):

   Scanner s = **new** Scanner(url.openStream(), "UTF-8")

3. Sie dürfen beim **Einlesen der Namen und Länder der IATA-Flughafentabellen** mit der ***replace***-Methode der Klasse ***Matcher*** arbeiten. ***Replace*** ersetzt alle Stellen einer Zeichenkette, die mit einem regulären Ausdruck übereinstimmen, mit einer übergebenen Zeichenkette. Diese Methode eignet sich insbesondere um Formatzeichen und Formattags aus Zeichenketten zu entfernen. Mehr kann ich nicht verraten.

4. **An allen anderen Stellen** ist die Verwendung der ***replace*** Methoden nicht erlaubt!

5. Stellen Sie den Zeichensatz in Eclipse auf UTF-8 um, damit alle Zeichen korrekt dargestellt werden. **Window → Preferences → General → Workspace**



Erweiterungen:

Nach erfolgreicher Bearbeitung zeigen wir Ihnen, wie Sie einzelne Webseiten mit der Lösung auswerten können!