Java XML kezelés labor

Készítette: Goldschmidt Balázs, BME IIT, 2015.

A feladatok megoldásához felhasználandó osztályok leírásait az alábbi URL-en találja meg:

https://docs.oracle.com/javase/8/docs/api/javax/xml/parsers/package-summary.html
https://docs.oracle.com/javase/8/docs/api/org/xml/sax/package-summary.html
http://www.jdom.org/docs/apidocs/

Alapfeladat

Készítsünk olyan Java alkalmazást, amely feldolgoz egy XML fájlt! A honlapon megtalálható XML fájl az OpenStreetMap BME körüli területét leíró adatait tartalmazza.

1 Egyszerű beolvasás SAX-szal

Írjunk Java programot, amely egy SAX parser segítségével a honlapon megadott XML fájlban megszámolja az XML tag-eket. Az eredményt tag-enként külön-külön adjuk meg, pl:

```
osm: 1
node: 3421
tag: 6893
way: 4581
...
```

Tippek

- A feladat megoldásához származtassuk az osztályunkat (*class TagCounter*) a *org.xml.sax.helpers.DefaultHandler* osztályból.
- A számoláshoz definiáljuk felül a *DefaultHandler startElement(...)* metódust!
- A program az XML fájl nevét az *-i* parancssori argumentum után kapja, pl:

```
java SaxTask -i bme.xml
```

- Legegyszerűbb az xml fájlt az eclipse projekt gyökerébe tenni, mert akkor a fenti relatív útvonallal lehet rá hivatkozni.
- A beolvasás elvégzéséhez a *main* metódusban használhatjuk az alábbi kódrészletet (*filename* az xml fájl elérési útja):

```
DefaultHandler h = new TagCounter();
SAXParserFactory factory = SAXParserFactory.newInstance();
 try {
     SAXParser p = factory.newSAXParser();
     p.parse(new java.io.File(filename), h);
 } catch (Exception e) {e.printStackTrace();}
```

2 Egyszerű adatfeldolgozás SAX-szal

Az előző feladatban elkészített alkalmazást módosítsuk úgy, hogy ki tudjuk szűrni a buszmegállókat!

Nézzük meg az XML fájlt! A fájlban az **<osm>** gyökértag alatt ***<node>*** tag-ek vannak, amik a térkép jellemző pontjait írják le. A buszmegállók tipikusan az alábbi struktúrában vannak megadva:

```
<node id="603263405" visible="true" version="13" changeset="28588770"
timestamp="2015-02-03T13:04:17Z"    user="itamas80"    uid="1719518"
lat="47.4740451" lon="19.0484570">
  <tag k="highway" v="bus_stop"/>
  <tag k="name" v="Újbuda-központ M"/>
  <tag k="old_name" v="Fehérvári út (Bocskai út)"/>
  <tag k="operator" v="BKV"/>
  <tag k="public_transport" v="stop"/>
  <tag k="ref:bkv" v="F02000"/>
  <tag k="verified" v="no"/>
  <tag k="wheelchair" v="limited"/>
 </node>
```

A program a feldolgozás során minden buszmegállónál írja ki, hogy mi a buszmegálló neve (zárójelben a régi neve) és hogy alkalmas-e kerekesszékes megközelítésre:

```
Megálló:
      Név: Újbuda-központ M (Fehérvári út (Bocskai út))
      Kerekesszék: limited
```

Tippek:

- A fenti adatok tárolásához szükség lesz egy *BusStop* osztályra, aminek attribútumaiba tudjuk az egyes elemi adatokat berakni (*name*, *oldName*, *wheelchair*).
- A feldolgozáshoz érdemes egyszerű állapotgépet implementálni a következő eseményekre:
  - ***<node>*** *kezdődik*: új buszmegálló-objektum (*BusStop*) létrehozása, az érvényesség *false*-ra állítása (ehhez a *BusStop*-ba kell egy *boolean valid* attribútum is).
  - ***<tag>*** *kezdődik, a "v" attribútum "bus_stop"*: az aktuális buszmegálló objektum *valid* mezőjét *true*-ra állítjuk. (*attributes.getValue("v")*)
  - ***<tag>*** *kezdődik, a* ***k*** *attribútum valamelyik várt értéket tartalmazza*: a megfelelő értéket beállítjuk.
  - ***<node>*** *végződik*: ha az aktuális buszmegálló objektum érvényessége *true*, kiírjuk az adatait
- Az xml fájl tag-jeinek (<node>, <tag>, stb) nevét a *startElement/endElement* metódusok *qName* paraméterben kapjuk meg, az XML tag-ek attribútumaihoz az *attributes* paraméterben férünk hozzá.

3 Összetett adatfeldolgozás SAX-szal

Az előző feladatban elkészített programot módosítsuk úgy, hogy a buszmegállókat a parancssorban megadott koordinátájú ponttól (pl. *-lat 47.4786346 -lon 19.0555773*) való távolságuk fordított sorrendjében írja ki!

Az egyes buszmegállók mellé a fentieken kívül írjuk ki azt is, hogy légvonalban milyen messze vannak a megadott ponttól.

Tippek:

- A megoldáshoz a *BusStop* osztályba fel kell venni egy *distance* attribútumot is a távolság tárolásához, amit a *startElement* metódusban kell beállítani, ha <*node*> tag érkezik.

- Az érvényes *BusStop*-okat egy listába érdemes gyűjteni (*endElement* metódusban a korábbi kiíratás helyett). Ezt a listát a beolvasás után távolság szerint növekvő módon rendezve kell kiíratni.

- A távolság kiszámításához használhatjuk az alábbi két számítás bármelyikét. Az első változat pontosabb eredményt ad.

```
double dist1(double lat1, double lon1, double lat2, double lon2) {
	double R = 6371000; // metres
	double phi1 = Math.toRadians(lat1);
	double phi2 = Math.toRadians(lat2);
	double dphi = phi2-phi1;
	double dl = Math.toRadians(lon2-lon1);

	double a = Math.sin(dphi/2) * Math.sin(dphi/2) +
		 Math.cos(phi1) * Math.cos(phi2) *
		 Math.sin(dl/2) * Math.sin(dl/2);
	double c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1-a));
	double d = R * c;
	return d;
}
double dist2(double lat1, double lon1, double lat2, double lon2) {
	double R = 6371000; // gives d in metres
	double phi1 = Math.toRadians(lat1);
	double phi2 = Math.toRadians(lat2);
	double dl = Math.toRadians(lon2-lon1);

	double d = Math.acos( Math.sin(phi1)*Math.sin(phi2)
		 + Math.cos(phi1)*Math.cos(phi2) * Math.cos(dl)
		) * R;
	return d;
}
```

4 Egyszerű beolvasás JDOM-mal

Írjon Java programot, amely a JDOM könyvtár segítségével a mellékelt XML fájlban megszámolja az XML tag-eket. A program az XML fájl nevét az *-i* parancssori argumentum után kapja, pl:

```
java JDomTask -i bme.xml
```

A beolvasáskor felépülő JDOM struktúrán kell úgy végigmennünk, hogy minden XML tag elemet érintsünk. A beolvasáshoz válasszuk a *SAXParser*-t.

A JDOM használatához

- töltsük le a labor honlapjáról a JDOM jar-t
- az Eclipse-ben jobb egérkattintással nyissuk meg a projekt *property*-ablakát
- válasszuk a *Java Build Path* menüt, ebben pedig a *Libraries* fület
- klikkeljünk az *Add external jars* gombra, és válasszuk ki a lementett JDOM jar-t.

5 Egyszerű adatfeldolgozás JDOM-mal

Az előző feladatban elkészített JDOM alapú alkalmazást módosítsuk úgy, hogy ki tudjuk válogatni a buszmegállókat!

A SAX megoldással szemben most a feladat a JDOM szerkezet módosítása. Az <osm> gyökérelem minden olyan gyerekét (és annak leszármazottait) törölni kell, amely nem buszmegállót tartalmazó <node>[1]. Használjuk az *Element* osztály *detach()* metódusát, de figyeljünk arra, hogy a *detach* azt a listát módosítja, amiben a szülőelem gyerekei vannak.

Az eredményül kapott DOM modellt írassuk ki a parancssori paraméterként (*-o* kapcsoló) megadott fájlba! Pl:

```
java DomTask -i bme.xml -o bme_bus.xml
```

[1] vagyis az Element vagy nem <node>, vagy nincs olyan <tag> gyereke, amelyben a "v" attribútum "bus_stop" értékű.

6 Összetett adatfeldolgozás JDOM-mal

Az előző feladatban elkészített programot módosítsuk úgy, hogy a buszmegállókat tartalmazó *<node>*-ok *<tag>* gyerekei közé felvesszük a parancssorban megadott koordinátájú ponttól (pl. -lat 47.4786346 -lon 19.0555773) számított távolságukat, pl:

```
<tag k="distance" v="324.123"/>
```

Tippek:

- A megoldáshoz új *Element* objektumot kell készíteni *"tag"* névvel.
- Az új *Element*-ben be kell állítani a *"k"* és *"v"* attribútumok értékét. A *"k"* attribútum vegye fel a *"distance"* értéket, a *"v"* attribútum pedig a számított távolságot (*Element.setAttribute(String, String)* metódus). A távolság kiszámításához használhatjuk a korábbi két számítás valamelyikét.
- Végül az új, módosított *Element*-et hozzá kell adni a megfelelő <node> szülőhöz (*Element* osztály *addContent(...)* metódusa).
- A kiíratáshoz használjuk az *XMLOutputter* osztály *output* metódusát!