Jena

Read

- The Jena RDF Tutorial[1]

This week's exercises will give you the first looks on the Jena API. Use the book to look for examples, but look also at the Jena Framework javadoc[2] , there might be better ways to do things than what is used in the book.

1   Creating RDF models with Jena

In this set of exercises we will learn how to create models with Jena. First, we will create the model "by hand", i.e., writing triples like in last week's exercises, then we will do the same using a Jena java program.

1.1   Exercise

Write an RDF representation of the graph in Figure 1. Add also some of the names of the the resources in the graphs in different languages, e.g., "Cat" spells "Katt" in Norwegian and "Katze" in German.

Use whatever namespace you prefer. The important thing is that you get a good idea of how the file should look like as you will need that in the next exercise.

1.2   Exercise

Write a java program which creates the same RDF model as you made in the previous exercise and writes the model to a file in the same RDF serialisation language as you used.

2   RDF serialisation converter

RDF may be written in different ways, in different serialisations. The common serialisations are RDF/XML, Turtle and N3.

---

[1] http://jena.apache.org/tutorials/rdf_api.html
[2] http://incubator.apache.org/jena/documentation/javadoc/jena/
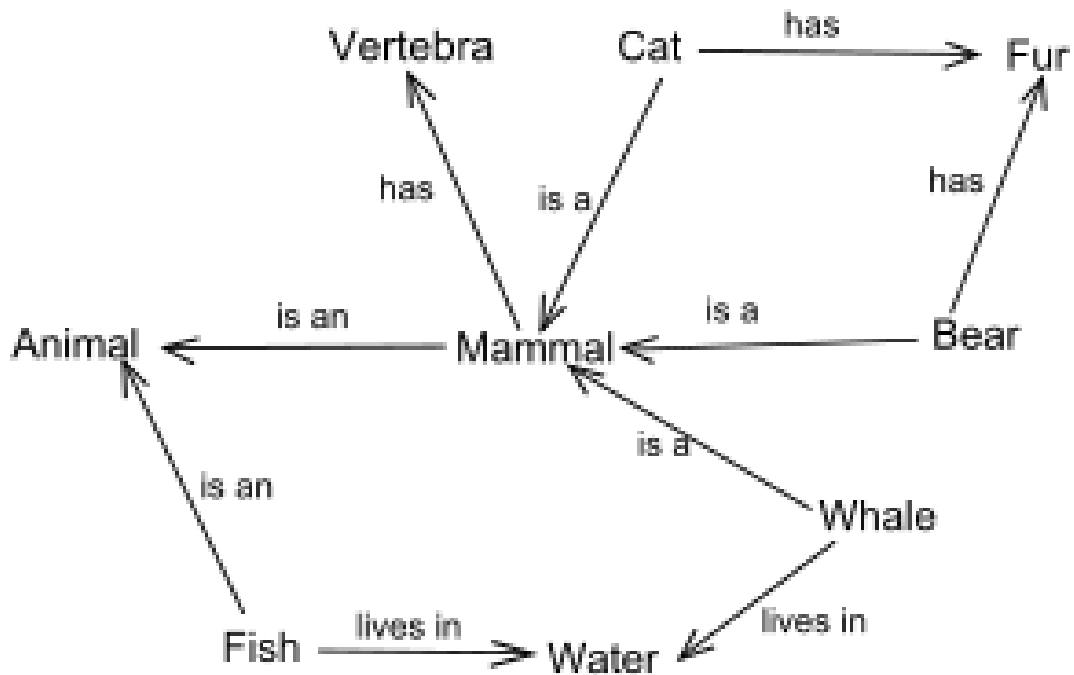
Figure 1: Graphs are suitable for encoding meaning.

2.1 Exercise

Write a Java program which can convert between four serialisations of RDF: RDF/XML, Turtle, N3 and N-TRIPLE. The program should take two arguments, input file and output format, where the file extension of the input file indicate which RDF syntax the file has and the output format specifies the format to convert to. The converted output shall be written to standard output. The common file extensions for RDF/XML, Turtle, N3 and N-TRIPLE are `.rdf`, `.ttl`, `.n3` and `.nt`, respectively.

Running

```
java your_java_program test.n3 RDF/XML > test.rdf
```

should convert the file `test.n3`, which is assumed written in N3, to RDF/XML format and write to the file `test.rdf`.

2.2 Exercise

Using your RDF syntax converter program, convert the Simpsons RDF graph you wrote in last week's exercise to **RDF/XML**.

2.3 Exercise

Using your RDF syntax converter program, convert the Simpsons RDF graph you wrote in last week's exercise to **Turtle**.

2.4 Exercise

Using your RDF syntax converter program, convert the Simpsons RDF graph you wrote in last week's exercise to **N3**.

2.5 Exercise

Using your RDF syntax converter program, convert the Simpsons RDF graph you wrote in last week's exercise to **N-Triples**.

2.6 Exercise

Explain the differences between the four RDF serialisations. Illustrate your points with examples from the RDF conversions done in these exercises.

See also http://www.w3.org/TR/turtle/ and http://www.w3.org/DesignIssues/Notation3.html.

3 Browsing RDF using Jena

In these exercises we will learn more about Jena by making a program that browses FOAF RDF graphs for friends.

3.1 Exercise

Make a java program which reads a FOAF file and lists the names of all the persons the FOAF person knows and the names of who they again know, i.e., all friends and the friends of these friends. The program should not use queries. You can assume that the name of a person is located in a `foaf:name` element and the foaf file of a person is located in a `rdfs:seeAlso` element—just like how your own FOAF file should look like if generated by the FOAF-a-Matic.

3.1.1 Tip

Running your program on Martin G. Skjæveland's FOAF file should give you something like this

```
Martin G. Skjæveland's friends:
   Tim Berners-Lee
   Martin Giese
      Kjetil Kjernsmo
      Arild Waaler
      Bruno Buchberger
      Reiner Hähnle
      David Norheim
      Renate Schmidt
      Roar Fjellheim
   Robert Engels
```

3.2 Exercise

Run your program on your own FOAF file.

4 RDF summary program

When opening OWL ontologies in ontology editors it is common to be presented with some metrics about the ontology, e.g., how many classes and relationships there are and what kinds of axioms are used. In this exercise we will create a similar program, but for RDF graphs.

4.1 Exercise

Write a program which reads an RDF file and outputs

- the total number of triples in the file

- the number of distinct subjects, predicates and objects in the graph

- all *types*, e.g., all objects in a triple where `rdf:type` is the predicate, and the distinct members of these types

4.1.1 Tip

Running your program with the following graph as input

```
1  @prefix :    <http://example.com/> .
2  @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
3
4   :m rdf:type :A .
5   :n rdf:type :A .
6   :m :r       :o .
7   :q :r       :n .
```

should give a result equivalent of the following:

```
RDF metrics: execs/summary_test.n3
No. of triples: 4
No. of subject: 3
No. of predicates: 2
No. of objects: 3

TYPES:
http://example.com/A
   http://example.com/n
   http://example.com/m
```

4.2 Exercise

Use your program to analyse your FOAF file.

4.3 Exercise

Use your program to analyse Martin Giese's FOAF file.

4.4 Exercise

Use your program to analyse your Simpsons RDF file.