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.

Solution

I have translated the edges labelled "is a" to `rdfs:subClassOf` predicates. (Don't worry, we'll learn about RDFS later) and otherwise kept the names of the edges and nodes as they are. I have added "Bjørn" and "Bear" as names of the resource `:Bear` using the predicate `rdfs:label`.

```
 1  @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
 2  @prefix : <http://www.example.org#> .
 3
 4  :Fish   rdfs:subClassOf :Animal ;
 5    :livesIn        :Water .
 6
 7  :Mammal rdfs:subClassOf :Animal ;
 8    :has            :Vertebra .
 9
10  :Whale rdfs:subClassOf :Mammal ;
11         :livesIn        :Water .
12
13  :Cat   rdfs:subClassOf :Mammal ;
```

[1] http://jena.apache.org/tutorials/rdf_api.html
[2] http://incubator.apache.org/jena/documentation/javadoc/jena/

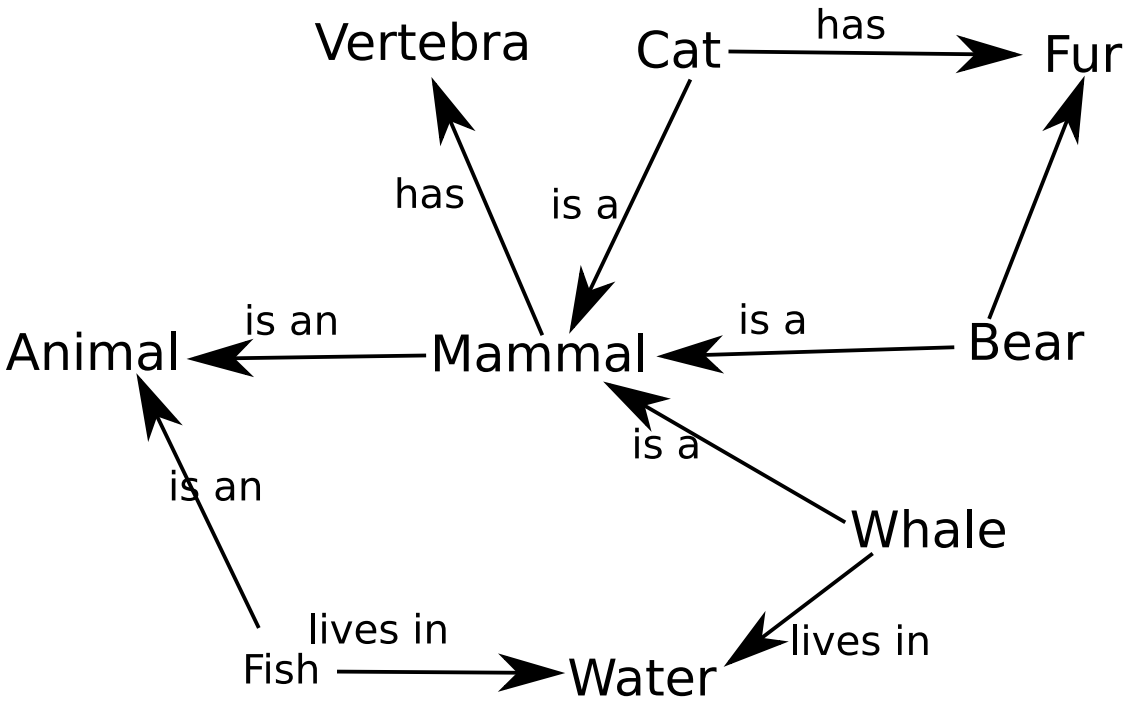

Figure 1: Graphs are suitable for encoding meaning.

```
14         :has            :Fur .
15
16  :Bear  rdfs:subClassOf :Mammal ;
17         :has            :Fur ;
18         rdfs:label      "Bjørn"@no ;
19         rdfs:label      "Bear"@en .
```

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.

Solution

My program is called `RDFAnimals`. The package `org.apache.jena.vocabulary.RDFS` contains static variables for the RDFS vocabulary. The animal model will be assembled in the variable `model`.

```
1  import org.apache.jena.rdf.model.*;
2  import org.apache.jena.vocabulary.RDFS;
3  public class RDFAnimals{
4  private Model model;
```

`writeModel`. Writes `model` to `System.out` in Turtle (TTL) format.

```
5  public void writeModel(){
6      model.write(System.out, "TTL");
7  }
```

`addStatement` takes three strings as arguments, which is assumed to be three URIs which constitute a triple, and creates respectively a subject, a predicate and an object of these strings. The subject, predicate and object is assembled to a statement (triple) and added to the `model`.

Note that this method assumes that all objects are resources and not literals, which I thought was OK for this example, since the program is custom built to create just this one RDF graph. A proper `addStatement` method should be more generic than my method.

```
 8  public void addStatement(String s, String p, String o){
 9    Resource subject = model.createResource(s);
10    Property predicate = model.createProperty(p);
11    RDFNode object = model.createResource(o);
12    Statement stmt = model.createStatement(subject, predicate, object);
13    model.add(stmt);
14  }
```

`createModel` creates an empty model in the variable `model`, defines the two namespaces I will be using, and uses the `addStatement` method to add triples to the model. Finally, "Bjørn" and "Bear" are added as `rdfs:label`-s to `:Bear`.

```
15  public void createModel(){
16    model = ModelFactory.createDefaultModel();
17    String ns = "http://www.example.org#";
18    String nsRDFS = "http://www.w3.org/2000/01/rdf-schema#";
19
20    addStatement(ns+"Fish", nsRDFS+"subClassOf", ns+"Animal");
21    addStatement(ns+"Fish", ns+"livesIn", ns+"Water");
22    addStatement(ns+"Mammal", nsRDFS+"subClassOf", ns+"Animal");
23    addStatement(ns+"Mammal", ns+"has", ns+"Vertebra");
24    addStatement(ns+"Whale", nsRDFS+"subClassOf", ns+"Mammal");
25    addStatement(ns+"Whale", ns+"livesIn", ns+"Water");
26    addStatement(ns+"Cat", nsRDFS+"subClassOf", ns+"Mammal");
27    addStatement(ns+"Cat", ns+"has", ns+"Fur");
28    addStatement(ns+"Bear", nsRDFS+"subClassOf", ns+"Mammal");
29    addStatement(ns+"Bear", ns+"has", ns+"Fur");
30
31    Resource bear = model.getResource(ns+"Bear");
32    bear.addProperty(RDFS.label, "Bjørn", "no");
33    bear.addProperty(RDFS.label, "Bear", "en");
34  }
```

`main` method. Creates the model and writes the model to standard out.

```
35    public static void main(String[] args){
36      RDFAnimals dave = new RDFAnimals();
37      dave.createModel();
38      dave.writeModel();
39    }
40  } // end RDFAnimals
```

Results

The result of running my program:

```
<http://www.example.org#Bear>
      <http://www.w3.org/2000/01/rdf-schema#label>
              "Bear"@en , "Bjørn"@no ;
      <http://www.w3.org/2000/01/rdf-schema#subClassOf>
              <http://www.example.org#Mammal> ;
      <http://www.example.org#has>
              <http://www.example.org#Fur> .
```

```
<http://www.example.org#Cat>
      <http://www.w3.org/2000/01/rdf-schema#subClassOf>
              <http://www.example.org#Mammal> ;
      <http://www.example.org#has>
              <http://www.example.org#Fur> .

<http://www.example.org#Whale>
      <http://www.w3.org/2000/01/rdf-schema#subClassOf>
              <http://www.example.org#Mammal> ;
      <http://www.example.org#livesIn>
              <http://www.example.org#Water> .

<http://www.example.org#Fish>
      <http://www.w3.org/2000/01/rdf-schema#subClassOf>
              <http://www.example.org#Animal> ;
      <http://www.example.org#livesIn>
              <http://www.example.org#Water> .

<http://www.example.org#Mammal>
      <http://www.w3.org/2000/01/rdf-schema#subClassOf>
              <http://www.example.org#Animal> ;
      <http://www.example.org#has>
              <http://www.example.org#Vertebra> .
```

2 RDF serialisation converter

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

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`.

Solution

A similar program for this is given in the book on page 82. See also http://jena.sourceforge.net/IO/iohowto.html.

My program is called `RDFConverter` and has three methods. `readModel` creates a model from a file. `writeModel` writes a model to file. The last method is `main`.

We start by importing the necessary libraries. `jena.rdf.model` is needed to make and write models, `jena.util` is used to create a model from file.

```
1  import org.apache.jena.rdf.model.*;
2  import org.apache.jena.util.*;
3  public class RDFConverter{
```

`readModel`. Creates a model from `file`.

```
4  public Model readModel(String file){
5    Model model = FileManager.get().loadModel(file);
6    return model;
7  }
```

`writeModel`. Writes `model` to `System.out` in the RDF syntax prescribed by `format`. If the RDF serialisation is specified as RDF/XML we use an RDFWriter to output the XML file using XML entities, thus increasing its readability.

```
 8  public void writeModel(Model model, String format){
 9    if(format.equals("RDF/XML")){
10      RDFWriter writer = model.getWriter(format);
11      writer.setProperty("showDoctypeDeclaration","true");
12      writer.write(model, System.out, null);
13    } else{
14      model.write(System.out, format);
15    }
16  }
```

`main`. Gets arguments from `args` and sends to other methods.

```
17    public static void main(String[] args){
18      String infile = args[0];
19      String outformat = args[1];
20
21      RDFConverter dave = new RDFConverter();
22      Model model = dave.readModel(infile);
23      dave.writeModel(model, outformat.toUpperCase());
24    }
25  } // end RDFConverter
```

2.2 Exercise

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

Results

```
<!DOCTYPE rdf:RDF [
  <!ENTITY foaf 'http://xmlns.com/foaf/0.1/'>
  <!ENTITY xsd 'http://www.w3.org/2001/XMLSchema#'>
  <!ENTITY rdf 'http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
  <!ENTITY sim 'http://www.ifi.uio.no/INF3580/simpsons#'>
  <!ENTITY fam 'http://www.ifi.uio.no/INF3580/family#'>]>
<rdf:RDF
    xmlns:rdf="&rdf;"
    xmlns:foaf="&foaf;"
    xmlns:fam="&fam;"
    xmlns:xsd="&xsd;"
    xmlns:sim="&sim;" >
  <rdf:Description rdf:about="&sim;Herb">
    <rdf:type rdf:resource="&foaf;Person"/>
```

```
</rdf:Description>
<rdf:Description rdf:about="&sim;Marge">
  <fam:hasSpouse rdf:nodeID="A0"/>
  <fam:hasSpouse rdf:resource="&sim;Homer"/>
  <foaf:name>Marge Simpson</foaf:name>
  <foaf:age rdf:datatype="&xsd;int">34</foaf:age>
  <rdf:type rdf:resource="&foaf;Person"/>
</rdf:Description>
<rdf:Description rdf:about="&sim;Lisa">
  <fam:hasParent rdf:nodeID="A1"/>
  <fam:hasParent rdf:nodeID="A2"/>
  <fam:hasMother rdf:resource="&sim;Marge"/>
  <fam:hasFather rdf:resource="&sim;Homer"/>
  <foaf:name>Lisa Simpson</foaf:name>
  <foaf:age rdf:datatype="&xsd;int">8</foaf:age>
  <rdf:type rdf:resource="&foaf;Person"/>
</rdf:Description>
<rdf:Description rdf:about="&sim;Patty">
  <fam:hasSister rdf:resource="&sim;Selma"/>
  <rdf:type rdf:resource="&foaf;Person"/>
</rdf:Description>
<rdf:Description rdf:about="&sim;Mona">
  <rdf:type rdf:resource="&foaf;Person"/>
</rdf:Description>
<rdf:Description rdf:nodeID="A2">
  <fam:hasSister rdf:resource="&sim;Selma"/>
  <fam:hasSister rdf:resource="&sim;Patty"/>
  <rdf:type rdf:resource="&foaf;Person"/>
</rdf:Description>
<rdf:Description rdf:nodeID="A3">
  <fam:hasMother rdf:resource="&sim;Mona"/>
  <rdf:type rdf:resource="&foaf;Person"/>
</rdf:Description>
<rdf:Description rdf:nodeID="A4">
  <fam:hasFather rdf:resource="&sim;Abraham"/>
  <rdf:type rdf:resource="&foaf;Person"/>
</rdf:Description>
<rdf:Description rdf:about="&sim;Simpsons">
  <fam:hasFamilyMember rdf:resource="&sim;Marge"/>
  <fam:hasFamilyMember rdf:resource="&sim;Bart"/>
  <fam:hasFamilyMember rdf:resource="&sim;Lisa"/>
  <fam:hasFamilyMember rdf:resource="&sim;Homer"/>
  <fam:hasFamilyMember rdf:resource="&sim;Maggie"/>
  <rdf:type rdf:resource="&fam;Family"/>
</rdf:Description>
<rdf:Description rdf:about="&sim;Maggie">
  <fam:hasParent rdf:nodeID="A3"/>
  <fam:hasParent rdf:nodeID="A4"/>
  <fam:hasMother rdf:resource="&sim;Marge"/>
  <fam:hasFather rdf:resource="&sim;Homer"/>
  <foaf:name>Maggie Simpson</foaf:name>
  <foaf:age rdf:datatype="&xsd;int">1</foaf:age>
  <rdf:type rdf:resource="&foaf;Person"/>
</rdf:Description>
<rdf:Description rdf:about="&sim;Selma">
  <fam:hasSister rdf:resource="&sim;Patty"/>
```

```
    <rdf:type rdf:resource="&foaf;Person"/>
  </rdf:Description>
  <rdf:Description rdf:about="&sim;Bart">
    <fam:hasMother rdf:resource="&sim;Marge"/>
    <fam:hasFather rdf:resource="&sim;Homer"/>
    <foaf:name>Bart Simpson</foaf:name>
    <foaf:age rdf:datatype="&xsd;int">10</foaf:age>
    <rdf:type rdf:resource="&foaf;Person"/>
  </rdf:Description>
  <rdf:Description rdf:about="&sim;Homer">
    <fam:hasSpouse rdf:resource="&sim;Marge"/>
    <foaf:name>Homer Simpson</foaf:name>
    <foaf:age rdf:datatype="&xsd;int">36</foaf:age>
    <rdf:type rdf:resource="&foaf;Person"/>
  </rdf:Description>
  <rdf:Description rdf:nodeID="A1">
    <fam:hasBrother rdf:resource="&sim;Herb"/>
    <rdf:type rdf:resource="&foaf;Person"/>
  </rdf:Description>
  <rdf:Description rdf:nodeID="A0">
    <fam:hasFather rdf:resource="&sim;Abraham"/>
    <rdf:type rdf:resource="&foaf;Person"/>
  </rdf:Description>
  <rdf:Description rdf:about="&sim;Abraham">
    <rdf:type rdf:resource="&foaf;Person"/>
  </rdf:Description>
</rdf:RDF>
```

2.3 Exercise

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

Results

```
@prefix foaf:    <http://xmlns.com/foaf/0.1/> .
@prefix xsd:     <http://www.w3.org/2001/XMLSchema#> .
@prefix rdf:     <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix sim:     <http://www.ifi.uio.no/INF3580/simpsons#> .
@prefix fam:     <http://www.ifi.uio.no/INF3580/family#> .

sim:Herb
      a       foaf:Person .

sim:Marge
      a       foaf:Person ;
      fam:hasSpouse sim:Homer ;
      fam:hasSpouse
              [ a       foaf:Person ;
                fam:hasFather sim:Abraham
              ] ;
      foaf:age "34"^^xsd:int ;
      foaf:name "Marge Simpson" .

sim:Lisa
      a       foaf:Person ;
```

```
      fam:hasFather sim:Homer ;
      fam:hasMother sim:Marge ;
      fam:hasParent
              [ a       foaf:Person ;
                fam:hasSister sim:Selma , sim:Patty
              ] ;
      fam:hasParent
              [ a       foaf:Person ;
                fam:hasBrother sim:Herb
              ] ;
      foaf:age "8"^^xsd:int ;
      foaf:name "Lisa Simpson" .

sim:Patty
      a       foaf:Person ;
      fam:hasSister sim:Selma .

sim:Mona
      a       foaf:Person .

sim:Simpsons
      a       fam:Family ;
      fam:hasFamilyMember sim:Marge , sim:Maggie , sim:Lisa , sim:Bart , sim:Homer .

sim:Maggie
      a       foaf:Person ;
      fam:hasFather sim:Homer ;
      fam:hasMother sim:Marge ;
      fam:hasParent
              [ a       foaf:Person ;
                fam:hasMother sim:Mona
              ] ;
      fam:hasParent
              [ a       foaf:Person ;
                fam:hasFather sim:Abraham
              ] ;
      foaf:age "1"^^xsd:int ;
      foaf:name "Maggie Simpson" .

sim:Selma
      a       foaf:Person ;
      fam:hasSister sim:Patty .

sim:Bart
      a       foaf:Person ;
      fam:hasFather sim:Homer ;
      fam:hasMother sim:Marge ;
      foaf:age "10"^^xsd:int ;
      foaf:name "Bart Simpson" .

sim:Homer
      a       foaf:Person ;
      fam:hasSpouse sim:Marge ;
      foaf:age "36"^^xsd:int ;
      foaf:name "Homer Simpson" .
```

```
sim:Abraham
      a       foaf:Person .
```

2.4 Exercise

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

Results

```
@prefix foaf:    <http://xmlns.com/foaf/0.1/> .
@prefix xsd:     <http://www.w3.org/2001/XMLSchema#> .
@prefix rdf:     <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix sim:     <http://www.ifi.uio.no/INF3580/simpsons#> .
@prefix fam:     <http://www.ifi.uio.no/INF3580/family#> .

sim:Herb
      a       foaf:Person .

sim:Marge
      a       foaf:Person ;
      fam:hasSpouse sim:Homer ;
      fam:hasSpouse
              [ a       foaf:Person ;
                fam:hasFather sim:Abraham
              ] ;
      foaf:age "34"^^xsd:int ;
      foaf:name "Marge Simpson" .

sim:Lisa
      a       foaf:Person ;
      fam:hasFather sim:Homer ;
      fam:hasMother sim:Marge ;
      fam:hasParent
              [ a       foaf:Person ;
                fam:hasSister sim:Selma , sim:Patty
              ] ;
      fam:hasParent
              [ a       foaf:Person ;
                fam:hasBrother sim:Herb
              ] ;
      foaf:age "8"^^xsd:int ;
      foaf:name "Lisa Simpson" .

sim:Patty
      a       foaf:Person ;
      fam:hasSister sim:Selma .

sim:Mona
      a       foaf:Person .

sim:Simpsons
      a       fam:Family ;
      fam:hasFamilyMember sim:Marge , sim:Maggie , sim:Lisa , sim:Bart , sim:Homer .

sim:Maggie
```

```
      a       foaf:Person ;
      fam:hasFather sim:Homer ;
      fam:hasMother sim:Marge ;
      fam:hasParent
              [ a       foaf:Person ;
                fam:hasMother sim:Mona
              ] ;
      fam:hasParent
              [ a       foaf:Person ;
                fam:hasFather sim:Abraham
              ] ;
      foaf:age "1"^^xsd:int ;
      foaf:name "Maggie Simpson" .

sim:Selma
      a       foaf:Person ;
      fam:hasSister sim:Patty .

sim:Bart
      a       foaf:Person ;
      fam:hasFather sim:Homer ;
      fam:hasMother sim:Marge ;
      foaf:age "10"^^xsd:int ;
      foaf:name "Bart Simpson" .

sim:Homer
      a       foaf:Person ;
      fam:hasSpouse sim:Marge ;
      foaf:age "36"^^xsd:int ;
      foaf:name "Homer Simpson" .

sim:Abraham
      a       foaf:Person .
```

2.5 Exercise

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

Results

These results are very verbose, but we include them for completeness.

```
_:AX2dX64cc37b2X3aX14b1609940cX3aXX2dX7ffd <http://www.ifi.uio.no/INF3580/family#hasSister> <http://www.ifi.uio.no/INF3580/simpsons#Selma> .
_:AX2dX64cc37b2X3aX14b1609940cX3aXX2dX7ffd <http://www.ifi.uio.no/INF3580/family#hasSister> <http://www.ifi.uio.no/INF3580/simpsons#Patty> .
_:AX2dX64cc37b2X3aX14b1609940cX3aXX2dX7ffd <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/foaf/0.1/Person> .
<http://www.ifi.uio.no/INF3580/simpsons#Patty> <http://www.ifi.uio.no/INF3580/family#hasSister> <http://www.ifi.uio.no/INF3580/simpsons#Selma> .
<http://www.ifi.uio.no/INF3580/simpsons#Patty> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/foaf/0.1/Person> .
<http://www.ifi.uio.no/INF3580/simpsons#Homer> <http://www.ifi.uio.no/INF3580/family#hasSpouse> <http://www.ifi.uio.no/INF3580/simpsons#Marge> .
<http://www.ifi.uio.no/INF3580/simpsons#Homer> <http://xmlns.com/foaf/0.1/name> "Homer Simpson" .
<http://www.ifi.uio.no/INF3580/simpsons#Homer> <http://xmlns.com/foaf/0.1/age> "36"^^<http://www.w3.org/2001/XMLSchema#int> .
<http://www.ifi.uio.no/INF3580/simpsons#Homer> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/foaf/0.1/Person> .
_:AX2dX64cc37b2X3aX14b1609940cX3aXX2dX7ffb <http://www.ifi.uio.no/INF3580/family#hasFather> <http://www.ifi.uio.no/INF3580/simpsons#Abraham> .
_:AX2dX64cc37b2X3aX14b1609940cX3aXX2dX7ffb <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/foaf/0.1/Person> .
<http://www.ifi.uio.no/INF3580/simpsons#Abraham> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/foaf/0.1/Person> .
<http://www.ifi.uio.no/INF3580/simpsons#Herb> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/foaf/0.1/Person> .
_:AX2dX64cc37b2X3aX14b1609940cX3aXX2dX7ffe <http://www.ifi.uio.no/INF3580/family#hasMother> <http://www.ifi.uio.no/INF3580/simpsons#Mona> .
_:AX2dX64cc37b2X3aX14b1609940cX3aXX2dX7ffe <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/foaf/0.1/Person> .
<http://www.ifi.uio.no/INF3580/simpsons#Selma> <http://www.ifi.uio.no/INF3580/family#hasSister> <http://www.ifi.uio.no/INF3580/simpsons#Patty> .
<http://www.ifi.uio.no/INF3580/simpsons#Selma> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/foaf/0.1/Person> .
_:AX2dX64cc37b2X3aX14b1609940cX3aXX2dX7ffc <http://www.ifi.uio.no/INF3580/family#hasBrother> <http://www.ifi.uio.no/INF3580/simpsons#Herb> .
_:AX2dX64cc37b2X3aX14b1609940cX3aXX2dX7ffc <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/foaf/0.1/Person> .
<http://www.ifi.uio.no/INF3580/simpsons#Lisa> <http://www.ifi.uio.no/INF3580/family#hasParent> _:AX2dX64cc37b2X3aX14b1609940cX3aXX2dX7ffc .
<http://www.ifi.uio.no/INF3580/simpsons#Lisa> <http://www.ifi.uio.no/INF3580/family#hasParent> _:AX2dX64cc37b2X3aX14b1609940cX3aXX2dX7ffd .
<http://www.ifi.uio.no/INF3580/simpsons#Lisa> <http://www.ifi.uio.no/INF3580/family#hasMother> <http://www.ifi.uio.no/INF3580/simpsons#Marge> .
<http://www.ifi.uio.no/INF3580/simpsons#Lisa> <http://www.ifi.uio.no/INF3580/family#hasFather> <http://www.ifi.uio.no/INF3580/simpsons#Homer> .
```

```
<http://www.ifi.uio.no/INF3580/simpsons#Lisa> <http://xmlns.com/foaf/0.1/name> "Lisa Simpson" .
<http://www.ifi.uio.no/INF3580/simpsons#Lisa> <http://xmlns.com/foaf/0.1/age> "8"^^<http://www.w3.org/2001/XMLSchema#int> .
<http://www.ifi.uio.no/INF3580/simpsons#Lisa> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/foaf/0.1/Person> .
<http://www.ifi.uio.no/INF3580/simpsons#Bart> <http://www.ifi.uio.no/INF3580/family#hasMother> <http://www.ifi.uio.no/INF3580/simpsons#Marge> .
<http://www.ifi.uio.no/INF3580/simpsons#Bart> <http://www.ifi.uio.no/INF3580/family#hasFather> <http://www.ifi.uio.no/INF3580/simpsons#Homer> .
<http://www.ifi.uio.no/INF3580/simpsons#Bart> <http://xmlns.com/foaf/0.1/name> "Bart Simpson" .
<http://www.ifi.uio.no/INF3580/simpsons#Bart> <http://xmlns.com/foaf/0.1/age> "10"^^<http://www.w3.org/2001/XMLSchema#int> .
<http://www.ifi.uio.no/INF3580/simpsons#Bart> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/foaf/0.1/Person> .
<http://www.ifi.uio.no/INF3580/simpsons#Maggie> <http://www.ifi.uio.no/INF3580/family#hasParent> _:AX2dX64cc37b2X3aX14b1609940cX3aXX2dX7ffe .
<http://www.ifi.uio.no/INF3580/simpsons#Maggie> <http://www.ifi.uio.no/INF3580/family#hasParent> _:AX2dX64cc37b2X3aX14b1609940cX3aXX2dX7fff .
<http://www.ifi.uio.no/INF3580/simpsons#Maggie> <http://www.ifi.uio.no/INF3580/family#hasMother> <http://www.ifi.uio.no/INF3580/simpsons#Marge> .
<http://www.ifi.uio.no/INF3580/simpsons#Maggie> <http://www.ifi.uio.no/INF3580/family#hasFather> <http://www.ifi.uio.no/INF3580/simpsons#Homer> .
<http://www.ifi.uio.no/INF3580/simpsons#Maggie> <http://xmlns.com/foaf/0.1/name> "Maggie Simpson" .
<http://www.ifi.uio.no/INF3580/simpsons#Maggie> <http://xmlns.com/foaf/0.1/age> "1"^^<http://www.w3.org/2001/XMLSchema#int> .
<http://www.ifi.uio.no/INF3580/simpsons#Maggie> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/foaf/0.1/Person> .
<http://www.ifi.uio.no/INF3580/simpsons#Simpsons> <http://www.ifi.uio.no/INF3580/family#hasFamilyMember> <http://www.ifi.uio.no/INF3580/simpsons#Marge> .
<http://www.ifi.uio.no/INF3580/simpsons#Simpsons> <http://www.ifi.uio.no/INF3580/family#hasFamilyMember> <http://www.ifi.uio.no/INF3580/simpsons#Bart> .
<http://www.ifi.uio.no/INF3580/simpsons#Simpsons> <http://www.ifi.uio.no/INF3580/family#hasFamilyMember> <http://www.ifi.uio.no/INF3580/simpsons#Lisa> .
<http://www.ifi.uio.no/INF3580/simpsons#Simpsons> <http://www.ifi.uio.no/INF3580/family#hasFamilyMember> <http://www.ifi.uio.no/INF3580/simpsons#Homer> .
<http://www.ifi.uio.no/INF3580/simpsons#Simpsons> <http://www.ifi.uio.no/INF3580/family#hasFamilyMember> <http://www.ifi.uio.no/INF3580/simpsons#Maggie> .
<http://www.ifi.uio.no/INF3580/simpsons#Simpsons> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.ifi.uio.no/INF3580/family#Family> .
<http://www.ifi.uio.no/INF3580/simpsons#Marge> <http://www.ifi.uio.no/INF3580/family#hasSpouse> _:AX2dX64cc37b2X3aX14b1609940cX3aXX2dX7ffb .
<http://www.ifi.uio.no/INF3580/simpsons#Marge> <http://www.ifi.uio.no/INF3580/family#hasSpouse> <http://www.ifi.uio.no/INF3580/simpsons#Homer> .
<http://www.ifi.uio.no/INF3580/simpsons#Marge> <http://xmlns.com/foaf/0.1/name> "Marge Simpson" .
<http://www.ifi.uio.no/INF3580/simpsons#Marge> <http://xmlns.com/foaf/0.1/age> "34"^^<http://www.w3.org/2001/XMLSchema#int> .
<http://www.ifi.uio.no/INF3580/simpsons#Marge> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/foaf/0.1/Person> .
<http://www.ifi.uio.no/INF3580/simpsons#Mona> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/foaf/0.1/Person> .
_:AX2dX64cc37b2X3aX14b1609940cX3aXX2dX7fff <http://www.ifi.uio.no/INF3580/family#hasFather> <http://www.ifi.uio.no/INF3580/simpsons#Abraham> .
_:AX2dX64cc37b2X3aX14b1609940cX3aXX2dX7fff <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/foaf/0.1/Person> .
```

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.

Solution

All languages are able to express RDF 1.1 according to the standard, so in that sense it does not matter which one we choose.

The smallest language of the four is N-Triples. It has no "syntactic sugar" to express similar triples, only "raw" triples. It is therefore very verbose, thus time consuming for humans to read or write, but very easy for computers to parse.

The full N3 language is more expressive than what is necessary for representing RDF. It has constructs for rules and for quantifiers, i.e., `@forAll` and `@forSome`.

Turtle lies in the sweet spot between N-Triples and N3, at least when for the purpose of expressing RDF. See http://www.w3.org/TeamSubmission/turtle/ chapters 8 and 9 for a comparison of the three. Comparing the output of the N3 simpsons file against the Turtle version the only relevant difference I could find between the two was that in N3 `rdf:type` is abbreviated to `a`, while Turtle does not use this abbreviation. This is probably due to the implementation in Jena, since the documentation of Turtle I have found allows this abbreviation.

3 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.

3.1 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

3.1.1 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
```

Solution

My class is called `RDFMetrics`. I include the necessary libraries (I'll come back to the ones we have not seen before) and make a constant for indenting the output.

```
 1  import org.apache.jena.rdf.model.*;
 2  import org.apache.jena.util.FileManager;
 3  import org.apache.jena.vocabulary.RDF;
 4  import org.apache.jena.graph.Node;
 5  import org.apache.jena.util.iterator.ExtendedIterator;
 6  import java.util.Iterator;
 7
 8  public class RDFMetrics {
 9    protected static Model modelRDF;
10    public static final String INDENT = "   ";
```

`readModel` is the same as we have seen before.

The next method gets the size of an iterator. All of the things that this program will count is kept in iterators.

```
11  public void readModel(String file) {
12    modelRDF = FileManager.get().loadModel(file);
13  }
14  public int getIteratorSize(Iterator i){
15    int count = 0;
16    while(i.hasNext()){
17      count++;
18      i.next();
```

```
19    }
20    return count;
21  }
```

Putting the predicates in a set removes duplicates.

```
22  private Set<Property> listPredicates(Model model) {
23    Iterator<Statement> stmts = model.listStatements();
24    Set <Property> preds = new HashSet<>();
25
26    while(stmts.hasNext()) {
27      preds.add(stmts.next().getPredicate());
28    }
29
30    return preds;
31  }
```

Then follows a block of counting methods. All methods retrieves an iterator containing the things we want to counts and passes it to the iterator counter method. Special cases are made for null values. Notice the funny cast of null to an RDFNode in `countSubjects` and `countPredicates`. This is to make the call unambiguous, there is also a `listStatements` method defined for superinterface of Model.

```
32  public int countStatements(Resource s, Property p, RDFNode o, Model m){
33    if(s == null && p == null && m == null){
34      return getIteratorSize(m.listStatements());
35    } else{
36      return getIteratorSize(m.listStatements(s, p, o));
37    }
38  }
39  public int countSubjects(Resource s, Model m){
40    if(s == null){
41      return getIteratorSize(m.listSubjects());
42    } else{
43      return getIteratorSize(m.listStatements(s, null, (RDFNode)null));
44    }
45  }
46  public int countPredicates(Property p, Model m){
47    if(p == null){
48      return listPredicates(m).size();
49    } else{
50      return getIteratorSize(m.listStatements(null, p, (RDFNode)null));
51    }
52  }
53  public int countObjects(RDFNode o, Model m){
54    if(o == null){
55      return getIteratorSize(m.listObjects());
56    } else{
57      return getIteratorSize(m.listStatements(null, null, o));
58    }
59  }
```

The following method prints the types in the model and all the distinct members of the type. First, it gets all objects of the property `rdf:type`, i.e., all objects in triples where `rdf:type` is the predicate, and iterates through them. Notice that Jena has a vocabulary class for RDF. For each object, it writes the URI to `System.out`, and gets and writes all the subjects in triples where the predicate is `rdf:type` and `o` is the object.

```
60  public void printType(Model model){
61    NodeIterator ni = model.listObjectsOfProperty(RDF.type);
62    while(ni.hasNext()){
63      RDFNode o = ni.next();
64      System.out.println(o.toString());
65      ResIterator ri = model.listResourcesWithProperty(RDF.type, o);
66      while(ri.hasNext()){
67    System.out.println(INDENT + ri.next());
68      }
69    }
70  }
```

`printTripleData` prints data about triples by using the counting method above. Since we want to count everything we set no restrictions, i.e., pass null to all methods.

```
71  public void printTripleData(Model model){
72    System.out.println("No. of triples: " +
73              countStatements(null, null, null, model));
74    System.out.println("No. of subject: " +
75              countSubjects(null, model));
76    System.out.println("No. of predicates: " +
77              countPredicates(null, model));
78    System.out.println("No. of objects: " +
79              countObjects(null, model));
80  }
```

`main` reads arguments and calls the two print data methods.

```
81  public static void main(String[] args){
82    RDFMetrics dave = new RDFMetrics();
83    dave.readModel(args[0]);
84    System.out.println("RDF metrics: " + args[0]);
85    dave.printTripleData(modelRDF);
86    System.out.println("\nTYPES:");
87    dave.printType(modelRDF);
88    }
89  } // end class
```

3.2 Exercise

Use your program to analyse your FOAF file (from last week).

Results

```
RDF metrics: execs/foaf.rdf
No. of triples: 40
No. of subject: 7
No. of predicates: 24
No. of objects: 30

TYPES:
http://xmlns.com/foaf/0.1/Person
   http://folk.uio.no/martige/foaf.rdf#me
   http://www.w3.org/People/Berners-Lee/card#i
   http://folk.uio.no/martingi/foaf.rdf#me
   http://www.esis.no/data/robert.engels/foaf.rdf#me
http://xmlns.com/foaf/0.1/Organization
   http://folk.uio.no/martige/foaf.rdf#ifi
```

```
http://xmlns.com/foaf/0.1/PersonalProfileDocument
   http://folk.uio.no/martige/foaf.rdf
http://www.w3.org/2003/01/geo/wgs84_pos#Point
   142bdc5b:14b1609a24e:-7fff
```

3.3 Exercise

Use your program to analyse your Simpsons RDF file.

Results

```
RDF metrics: execs/simpsons.ttl
No. of triples: 52
No. of subject: 16
No. of predicates: 10
No. of objects: 27

TYPES:
http://www.ifi.uio.no/INF3580/family#Family
   http://www.ifi.uio.no/INF3580/simpsons#Simpsons
http://xmlns.com/foaf/0.1/Person
   http://www.ifi.uio.no/INF3580/simpsons#Herb
   http://www.ifi.uio.no/INF3580/simpsons#Marge
   http://www.ifi.uio.no/INF3580/simpsons#Lisa
   http://www.ifi.uio.no/INF3580/simpsons#Patty
   http://www.ifi.uio.no/INF3580/simpsons#Mona
   5060bfbb:14b1609aa67:-7ffb
   5060bfbb:14b1609aa67:-7ffc
   http://www.ifi.uio.no/INF3580/simpsons#Maggie
   5060bfbb:14b1609aa67:-7ffe
   http://www.ifi.uio.no/INF3580/simpsons#Bart
   5060bfbb:14b1609aa67:-7ffd
   http://www.ifi.uio.no/INF3580/simpsons#Selma
   http://www.ifi.uio.no/INF3580/simpsons#Homer
   5060bfbb:14b1609aa67:-7fff
   http://www.ifi.uio.no/INF3580/simpsons#Abraham
```