INFO-H509: XML & Web Technologies
Semantic Web Exercises

Lecturer: Stijn Vansummeren
Assistant: Michael Waumans

2015–2016

On the website you will find a set of RDF documents (in RDF/XML and Turtle format) that describe a simplified course catalog. They are necessary to do the following exercises.

1 RDF

Exercise 1.1

1. Inspect the contents of the file `staff.rdf` (this file is in the RDF/XML format). Draw the corresponding RDF graph on a sheet of paper.

   Note: You can verify the correctness of your solution by using the online tool available at `http://www.w3.org/RDF/Validator/`: copy the file's content and paste it in the text box. Make sure that under *Display Result Options*, the option *Triples and/or graph* is set to "Triples and Graph". (See Figure 1.)

2. Complete your graph drawing by adding the information about the course *XML Technologies* that you find in the file `catalogue.rdf`.

3. Further complete the graph on paper by taking into account the information from the file `infoh509.ttl`.

4. Based on the graph that you have drawn so far, write down the triples that you would need to record the information of the second lecture of the course *XML Technologies* in the graph.

Exercise 1.2

Open the file `catalog.rdf` with a text editor, and add to it a new course with the title "Object Oriented Programming, whose code is INFO-H-200. This course is lectured by prof. Zimányi and the assistant is called Boris Verhaegen

Next, modify the file `catalog.rdf` to record the fact that the course INFO-H-303 Databases is a prerequisite for the course *Distributed Information Systems*. (You can use the predicate `ulb:prerequisite` for this.)

Similarly, add the fact that the course INFO-H-100 is a prerequisite for *Object Oriented Programming*. (You can again use the predicate `ulb:prerequisite` for this.)

Exercice 1.3

Write a document using the `Turtle` syntax that describes who you are (in the same spirit as the information in the `staff.ttl` file). Indicate in this file that you follow the course *XML Technologies*. Any new terms that you may need to invent in order to do this should be added under `http://www.example.org/[your name]/`.

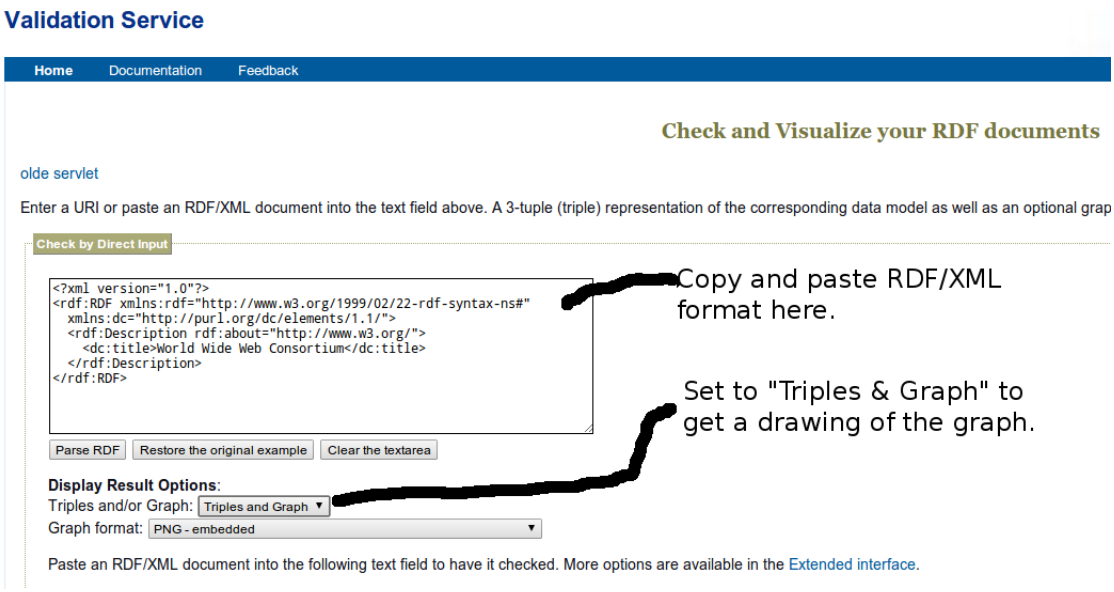

Figure 1: The RDF Validator Tool

2 RDF Schema

Exercise 2.1

The file `inference.ttl` contains an RDF Schema ontology that specifies that the typical people working at a university (`ulb:Professor`, `ulb:PHDStudent`) are members of the academic personnel (`ulb:Faculty`).

1. The data files available on the course webpage also contains a Java JAR that allows us to print out all RDF triples that can be inferred from a given set of input RDF documents. To show all triples that can be inferred when interpreting the RDF Schema information available, you can call this as follows:

   ```
   java -jar infertools.jar [--rdfs | --owlfull] <list of files containing rdf>
   ```

   - By default, `-owlfull` is assumed. In this mode, OWL inferences, including axiomatic triples, will be made. Note that the tool uses the Jena (`jena.apache.org`) library for OWL reasoning; the algorithm provided by this toolkit is not complete however, but should be complete for the OWL DL things that are to be used in these exercises.
   - The `-rdfs` option causes inferencing to be done only on the RDF schema vocabulary. Axiomatic triples are not output.

   Hence, from within a Windows Command Prompt or Linux Terminal in the directory where you extracted the data files, you would do the following command to make RDF schema inferences on staff.rdf using the rules in inference.ttl.

   ```
   java -jar infertools.jar --rdfs inference.ttl staff.rdf
   ```

2. Run this command and inspect the output. Notice that all professors and PhD students are now also classified as belonging to the class `ulb:Faculty`.

3. Modify the file `inference.ttl` by adding a rule that states that all personnel members (`ulb:Faculty`) are people (`foaf:Person`).

4. Run the above command again and inspect the output. Verify that your rule works as expected.

Exercise 2.2

At current, the `catalog.rdf` file does not specify any type for the courses *Databases* and *Introduction to Computer Programming*. Use the RDF Schema terms `rdfs:range` and `rdfs:domain` to add rules to the `inference.ttl` file that state that the prerequisite of a course is itself a course. Run the tool from exercise 2.2. again to verify that your addition is correct.

Supplementary exercise: In a similar vain, still using `rdfs:range` and `rdfs:domain`, add rules that describe the properties `lecturer` and `assistant` in more detail.

Exercice 2.3

Create a new property `workHomepage`, and specify that it is a sub property of foaf:homepage. Modify `staff.rdf` to use this property.

3 OWL

Note For these exercises, when calling `infertools.jar` be sure to call it with the `-owlfull` option to allow OWL inferences to be made.

Exercise 3.1

Recall that OWL DL defines the following property characteristics.

1. `owl:TransitiveProperty`
2. `owl:SymmetricProperty`
3. `owl:FunctionalProperty`
4. `owl:InverseFunctionalProperty`

For each of the following properties, list which of these property characteristics could apply.

1. The prerequisite of a course (`ulb:prerequisite`)
2. The student number of a student
3. Birthdate
4. owl:sameAs
5. owl:inverseOf

Supplementary exercise: Complete the description of `ulb:prerequisite` in the file `inference.ttl`. Verify the effect using the `inferencetool.jar` utility.

Exercise 3.2

Define the `ulb:teaches` property as the inverse of `ulb:lecturer`. Also define the domain and range of this property.

Exercise 3.3

Define that `staff:fpicalau` is the same person as `http://my.opera.com/fpicalausa/xml/foaf#me`.

Exercise 3.4

Use OWL DL to model the following sentences:

- The class `Vegetable` is a subclass of `PizzaTopping`.
- The class `PizzaTopping` does not share any elements with the class `Pizza`.
- The individual `aubergine` is an element of the class `Vegetable`.
- The abstract property is only used for relationships between elements of the classes `Pizza` and `PizzaTopping`.
- The class `VegPizza` consists of those elements which are in the class `NoMeatPizza` and in the class `NoFishPizza`.
- The property `hasTopping` is a subproperty of `hasIngredient`.

Next:

- Add an individual to both the class `PizzaTopping` and `Pizza`. Run the inference tool. What do you get? What should you do to remedy this?
- Add an individual to both the class `NoMeatPizza` and `NoFishPizza`. What do you expect to get? Run the inference tool. Verify using inference tool that you indeed get the expected result.

Exercise 3.5

Continuing Exercise 3.4, use OWL DL to model the following sentences.

1. Every pizza has `tomato` as a topping.
2. Every pizza in the class `PizzaMargarita` has exactly `tomato` and `cheese` as topping.

4 SPARQL

note The data files available on the course webpage also contains a Java JAR that allows us to execute a SPARQL query on a set of input RDF documents. You can call this as follows:

```
java -jar sparqltool.jar  data-files [query.sparql]
```

- Here, `data-files` is a white-space separated list of rdf files (recognized extensions: ttl, rdf, n3, nt).
- `query.sparql` is the optional filename with "sparql" as extension containing the query to execute.
- If no query file is specified, the query is read on standard input until double-return.

Exercise 4.1

Write SPARQL queries for the following queries. (All queries should be run with files `catalog.rdf`, `staff.rdf`, and `infoh509.ttl`.)

1. Retrieve the URIs of all the courses.
2. Retrieve, for each course, its title and the name of the lecturer.
3. The name of all Professors who teach a course such that (s)he also teaches a prerequisite for that course.
4. The name of all persons (`foaf:Person`) and their personal homepage, should this be available (i.e., retrieve just the person if no homepage is available).

5. The title of all courses that have been organized in *UA4.218*.

6. **Supplementary exercise:** All persons who know someone who know M.Vansummeren.