INM713 Semantic Web Technologies and Knowledge Graphs

Laboratory 2: Creating (Simple) RDF-based Knowledge Graphs

Ernesto Jiménez-Ruiz

Academic course: 2020-2021
Updated: January 27, 2021

Contents

1 Git Repositories

Support codes for the laboratory sessions are available in *github*. There are two repositories, one in Python and another in Java:

`https://github.com/city-knowledge-graphs`

2 FOAF - Friend of a friend

The FOAF project is an old project where RDF was the core technology. FOAF aimed at being a simple technology to make it easier to share and use information about people and their activities. See the following webpages for more information.

- FOAF project: `http://www.foaf-project.org/`
- Wikipedia: `https://en.wikipedia.org/wiki/FOAF_(ontology)`

Task 2.1: Go to the FOAF-a-Matic online service (`http://ldodds.com/foaf/foaf-a-matic.en.html`) and create your own FOAF file. Add myself as friend (*See Also* field: `https://sws.ifi.uio.no/vocab/ernesto_foaf.rdf`).

Task 2.2: If you have a personal Web space, try to publish the generated FOAF RDF file. Use the FOAF visualiser (`https://foaf-visualizer.gnu.org.ua/`) with your FOAF file or with mine: `https://foaf-visualizer.gnu.org.ua/?uri=https://sws.ifi.uio.no/vocab/ernesto_foaf.rdf`

Task 2.3. Load your (or my) FOAF with RDFLib or Jena API.

- Print the triples.
- Convert the RDF/XML file into RDF/Turtle.

FOAF was an interesting project, but there are currently more recent efforts. For example, Wikidata (`https://www.wikidata.org/`) is a community driven knowledge graph including both manually and automatically created entries. Check my entry: `https://www.wikidata.org/wiki/Q56614973`.

3 Creating an RDF graph

There are different options to create RDF triples:

- Using your favourite text editor.
- Using an ontology editor like Protégé.
- Programmatically with RDFlib (Python) or Jena API (Java).

Task 3.1: Use a text editor to create triples in Turtle format. Create:

- An entity that represents yourself of type `foaf:Person`.
- Triples for your name and surname.
- Triples for your city and country of birth.
- Triple(s) with the list of languages you speak.
- Triples describing your past or current employer/university, stating the date of start and end (if applicable).

Tips:

- Select a suitable namespace for your entities.
- Define prefixes.
- Reuse vocabulary if possible (*e.g.*, `http://dbpedia.org/resource/Spain`, `https://dbpedia.org/ontology/birthPlace`, `http://xmlns.com/foaf/0.1/name`).
- You may need to use reification or a n-ary relationship.
- Give a *.ttl* extension to your created file.

Task 3.2: Load the created .ttl file with RDFLib or Jena API and print the triples in the graph. This may give errors if the created triples have not been properly formatted.

4 Ontology editor Protégé: first steps

Protégé is an editor of OWL ontologies (`https://protege.stanford.edu/`). We will use the Desktop version. Protégé will also serve us to create triples.[1] For example to create the triples in Table 1, one needs to follow these steps:[2]

Table 1: Example RDF triples.

subject	predicate	object
city:ernesto	rdf:type	foaf:Person
city:ernesto	foaf:name	"Ernesto Jimenez-Ruiz"^^xsd:string
city:ernesto	city:teaches	city:INM713
city:INM713	rdf:type	city:Module

- Change *Ontology IRI* in "Active ontology/Annotations". *e.g.*, `http://www.semanticweb.org/inm713/lab2` (see Figure 1).
- Create *prefixes* in "Active ontology/Ontology Prefixes" (see Figure 2).

[1] In OWL we call them axioms. Not all axioms have a direct triple representation. In this session we will use *type definitions* and *property assertions* which have a direct triple serialization in RDF.

[2] Resulting triples are available in GitHub (`lab2_protege.ttl`)

  - city: `http://www.example.org/university/london/city#`
  - foaf: `http://xmlns.com/foaf/0.1/`

- Create *classes* Person and Module in Tab "Entities/Classes" (see Figure 3):
  - `city:Module`
  - `foaf:Person`
- Create an *object property* `city:teaches` in Tab "Entities/Object properties" (see Figure 4).
- Create a *data property* `foaf:name` in Tab "Entities/Data properties" (See Figure 5).
- Create *individuals* in Tab "Entities/Individuals" (see Figure 6).
  - `city:ernesto`
  - `city:inm713`
- Assign *types* to individuals. Select a type from class hierarchy (see Figure 7).
- Create an *Object property assertion* (*e.g.*, a triple with property `city:teaches`) associated to individual `city:ernesto` (see Figure 8).
- Create a *Data property assertion* (*e.g.*, a triple with property `foaf:name`) associated to individual `city:ernesto` (see Figure 9).

Task 4.1: Repeat Task 3.1 with Protégé.

- Tip 1: *Protégé (and OWL) does not allow the creation of blank nodes. Solutions involving reification or n-ary relationships can use a named individual.*
- Tip 2: *Protégé can however annotate axioms with annotation properties (i.e., no logical implications), see Figure 10.*
- Tip 3: *In OWL, instead of having a list of values, one should create different property assertions.*

5 Creating triples programatically

Both RDFlib and Jena API provide methods to populate an empty graph or add triples to an existing one.

Task 5.1: Repeat Task 3.1 with your favourite RDF library. Tip: *See examples in GitHub.*

RDFLib documentation:

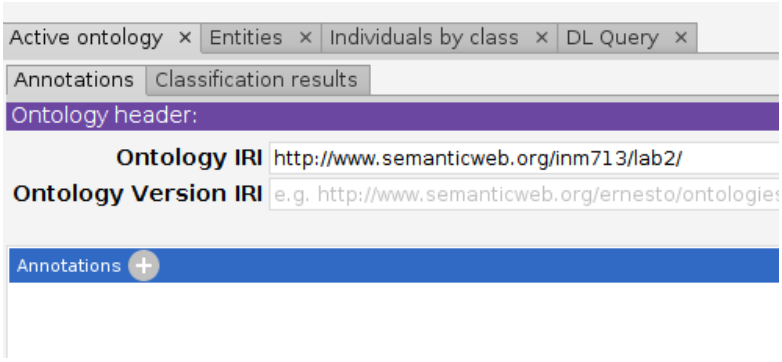

Figure 1: Changing ontology IRI in Protégé.

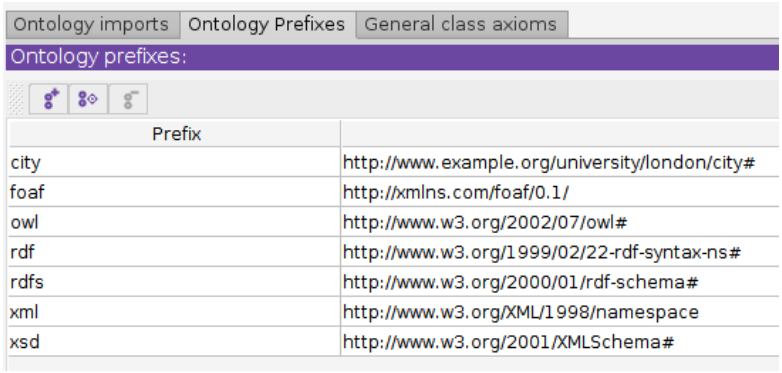

Figure 2: Creating prefixes in Protégé.

- `https://rdflib.readthedocs.io/en/stable/intro_to_creating_rdf.html`
- `https://rdflib.readthedocs.io/en/stable/rdf_terms.html`

JENA API documentation:

- `https://jena.apache.org/tutorials/rdf_api.html`
- `https://jena.apache.org/documentation/notes/`

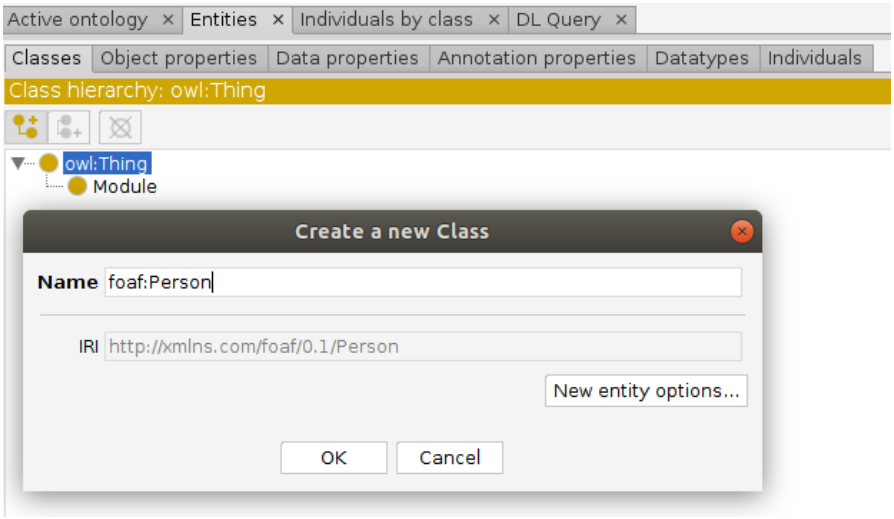

Figure 3: Creating classes in Protégé.

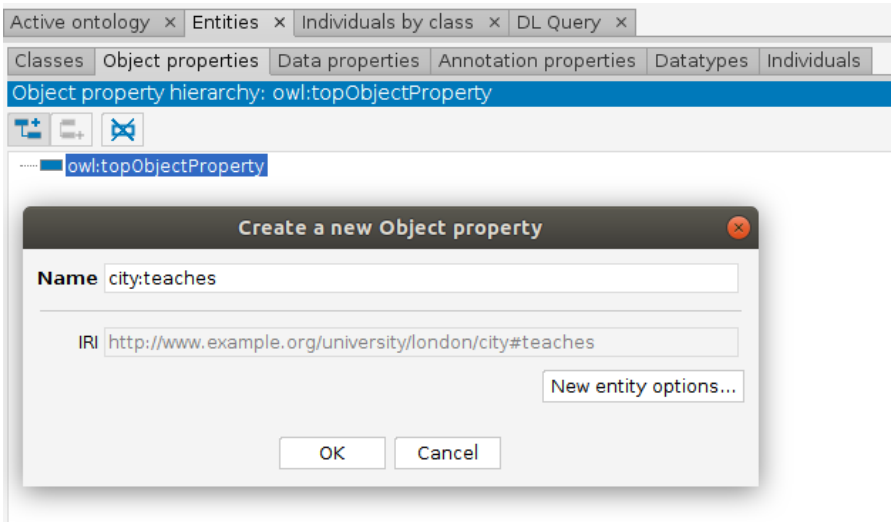

Figure 4: Creating an object property in Protégé.

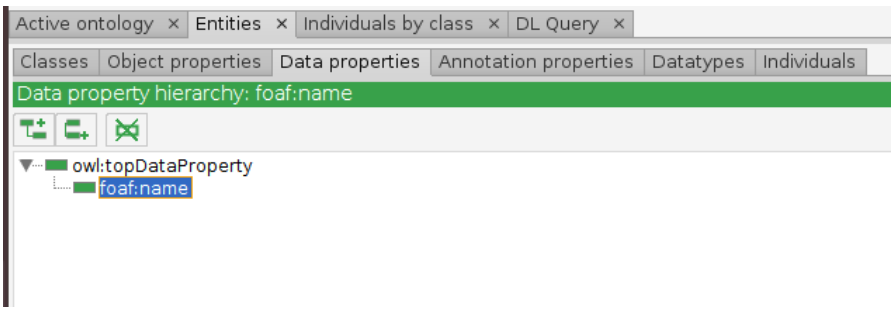

Figure 5: Creating a data property in Protégé.

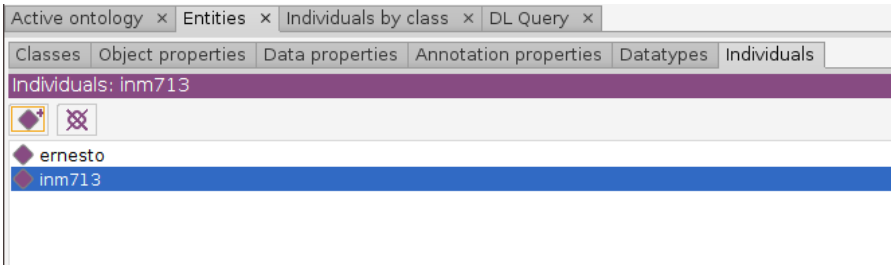

Figure 6: Creating individuals in Protégé.

Figure 7: Assigning types in Protégé.

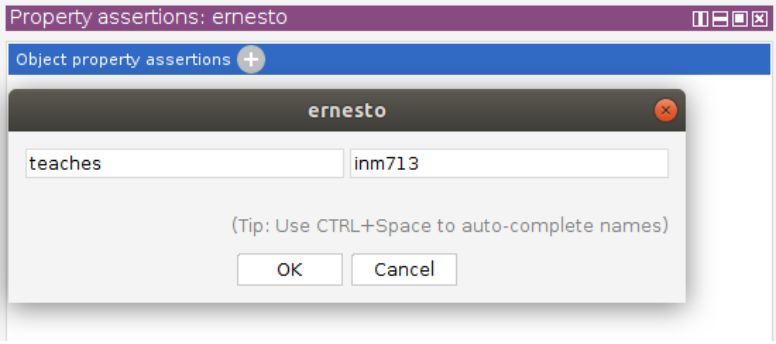

Figure 8: Creating triple with a object property in Protégé.

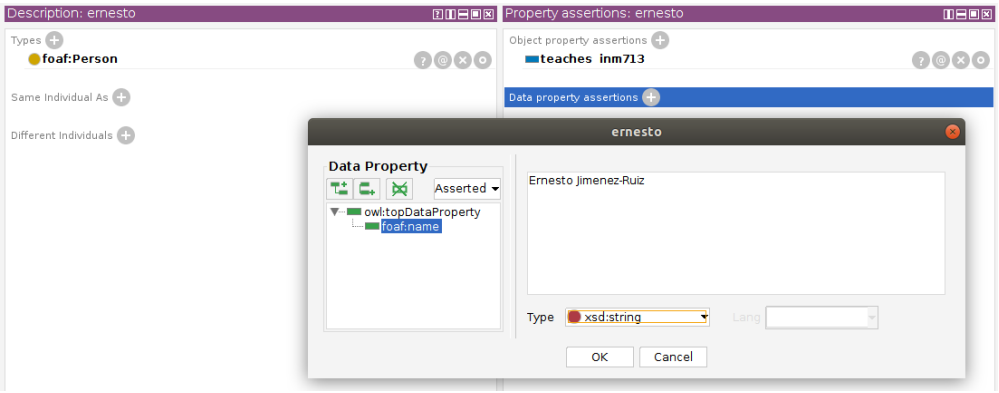

Figure 9: Creating triple with a data property in Protégé.

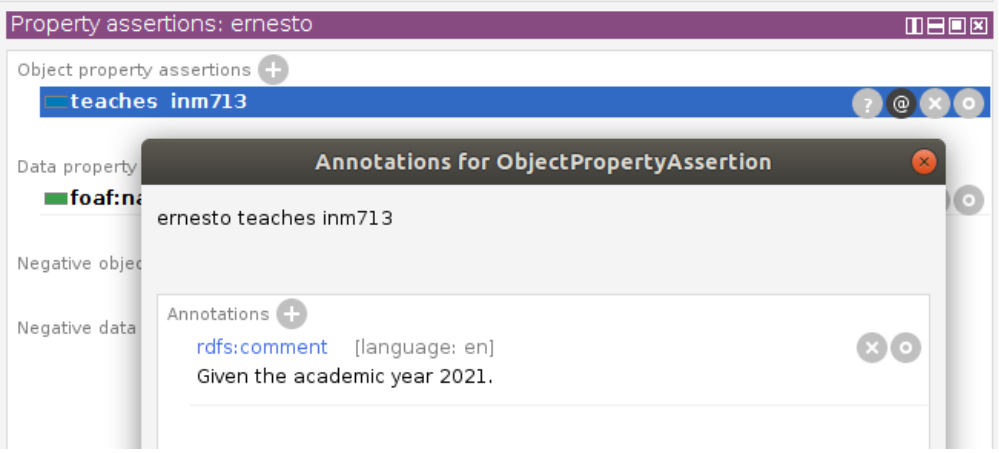

Figure 10: Creating annotation for a property assertion in Protégé.