Protégé + Fuseki + Jena-OWL + YASGUI

Installation and execution example

Semantic Web course 2013, Stefan Schlobach and Ronald Siebes

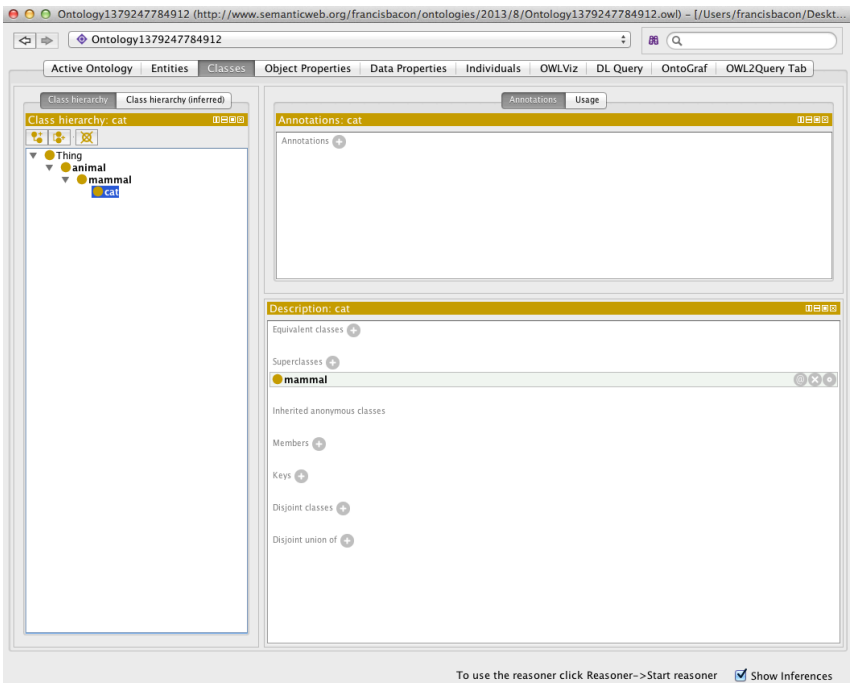
About

A manual to set up Fuseki with Jena-owl reasoning over an ontology created in Protégé and querying it via YASGUI.

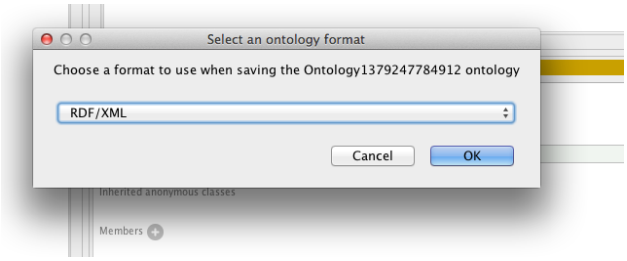
Software used in this example

Protégé Desktop 4.3: http://protege.stanford.edu/download/protege/4.3/zip/?C=M;O=D
Jena-Fuseki-0.2.7: http://www.apache.org/dist/jena/binaries/jena-fuseki-0.2.7-distribution.zip
YASGUI: http://yasgui.laurensrietveld.nl/

Step 1: create and save an ontology in Protégé



Create three classes, where mammal is subclass of animal and cat subclass of mammal



Save the ontology in RDF/XML format

Step 2: install, setup and run Fuseki

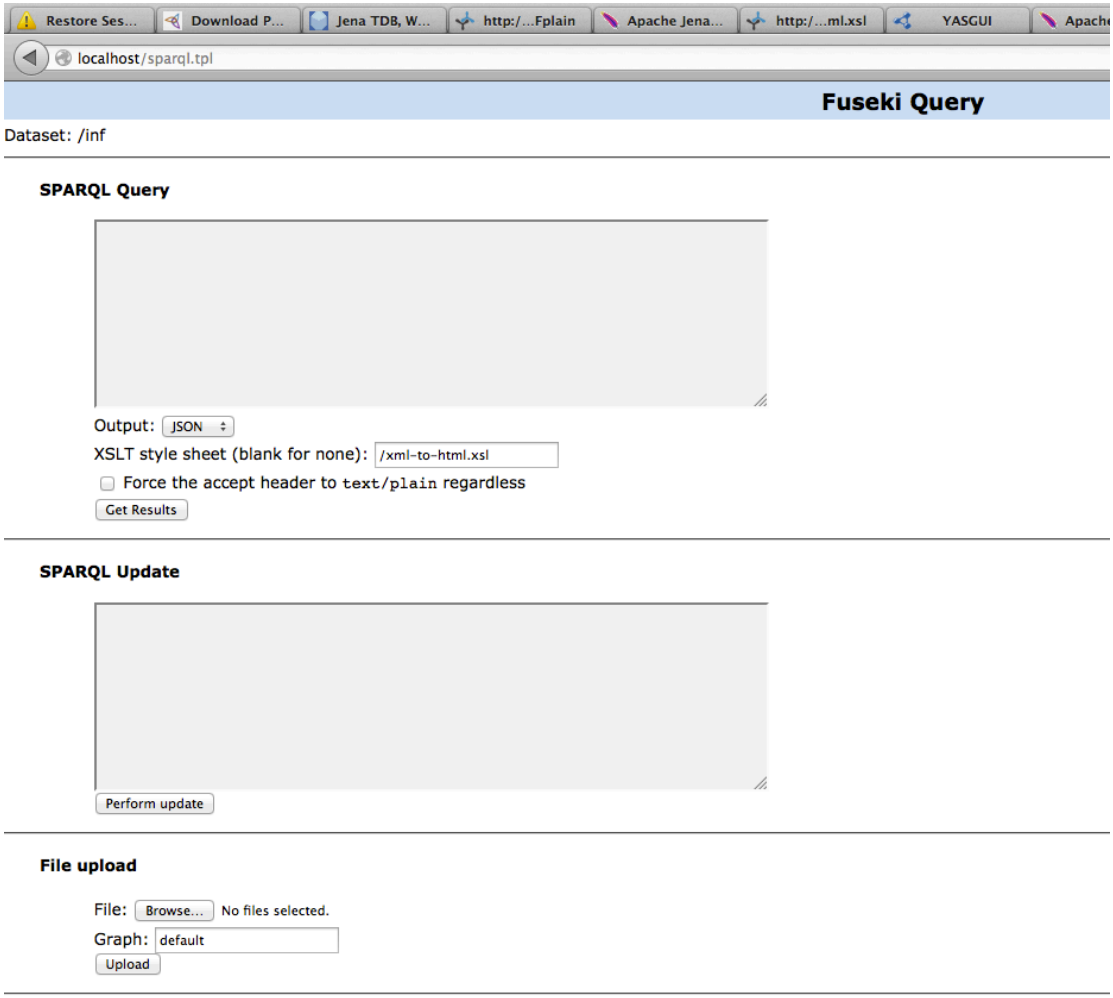
- download Jena-Fuseki
- unpack it
- go to the installation folder and open config-inf-tdf.ttl with a texteditor
- paste the following:

```
# Licensed under the terms of http://www.apache.org/licenses/LICENSE-2.0

@prefix :        <#> .
@prefix fuseki:  <http://jena.apache.org/fuseki#> .
@prefix rdf:     <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

@prefix rdfs:   <http://www.w3.org/2000/01/rdf-schema#> .
@prefix tdb:     <http://jena.hpl.hp.com/2008/tdb#> .
@prefix ja:      <http://jena.hpl.hp.com/2005/11/Assembler#> .

[] rdf:type fuseki:Server ;
   fuseki:services (
     <#service1>
   ) .

# Custom code.
[] ja:loadClass "com.hp.hpl.jena.tdb.TDB" .

# TDB
tdb:DatasetTDB  rdfs:subClassOf  ja:RDFDataset .
tdb:GraphTDB    rdfs:subClassOf  ja:Model .

## ---------------------------------------------------------------
## Service with only SPARQL query on an inference model.
## Inference model bbase data in TDB.

<#service1>  rdf:type fuseki:Service ;
    fuseki:name              "inf" ;             # http://host/inf
    fuseki:serviceQuery      "sparql" ;          # SPARQL query service
    fuseki:serviceUpdate     "update" ;
    fuseki:serviceUpload              "upload" ;   # Non-SPARQL upload service
    fuseki:serviceReadWriteGraphStore "data" ;     # SPARQL Graph store protocol (read and
write)
    # A separate ead-only graph store endpoint:
    fuseki:serviceReadGraphStore      "get" ;      # SPARQL Graph store protocol (read
only)y

    fuseki:dataset           <#dataset> ;
    .

<#dataset> rdf:type       ja:RDFDataset ;
    ja:defaultGraph       <#model_inf> ;
     .

<#model_inf> a ja:InfModel ;
     ja:baseModel <#tdbGraph> ;
     ja:reasoner [
         ja:reasonerURL <http://jena.hpl.hp.com/2003/OWLFBRuleReasoner>
     ] .

<#tdbDataset> rdf:type tdb:DatasetTDB ;
    tdb:location "DB" ;
    # If the unionDefaultGraph is used, then the "update" service should be removed.
    # The unionDefaultGraph is read only.
    # tdb:unionDefaultGraph true ;
    .

<#tdbGraph> rdf:type tdb:GraphTDB ;
    tdb:dataset <#tdbDataset> .
```

- execute (LINUX/MAC) "sudo ./fuseki-server --port=80 --update --config=config-inf-tdb.ttl" (port 80 is important for it to work with yasgui), or (WINDOWS) "fuseki-server.bat --port=80 --update --config=config-inf-tdb.ttl" (port 80 is important for it to work with yasgui)
- open http://localhost/control-panel.tpl in your browser

Step 3: load and query your ontology via SPARQL

- select the /inf dataset

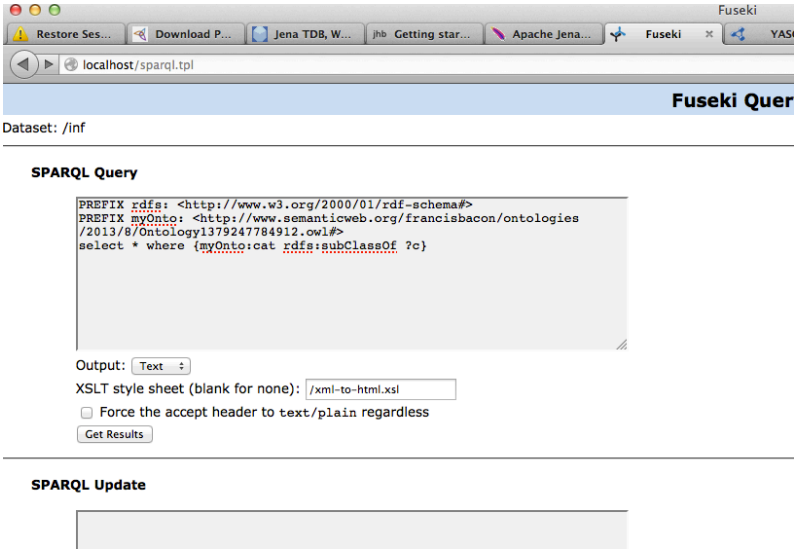


should look like this

- upload the ontology that you saved
- test it by typing the sparql query

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX myOnto:
<http://www.semanticweb.org/francisbacon/ontologies/2013/8/Ontology137
9247784912.owl#>
select * where {myOnto:cat rdfs:subClassOf ?c}
```

(note that your ontology will have a different URI)

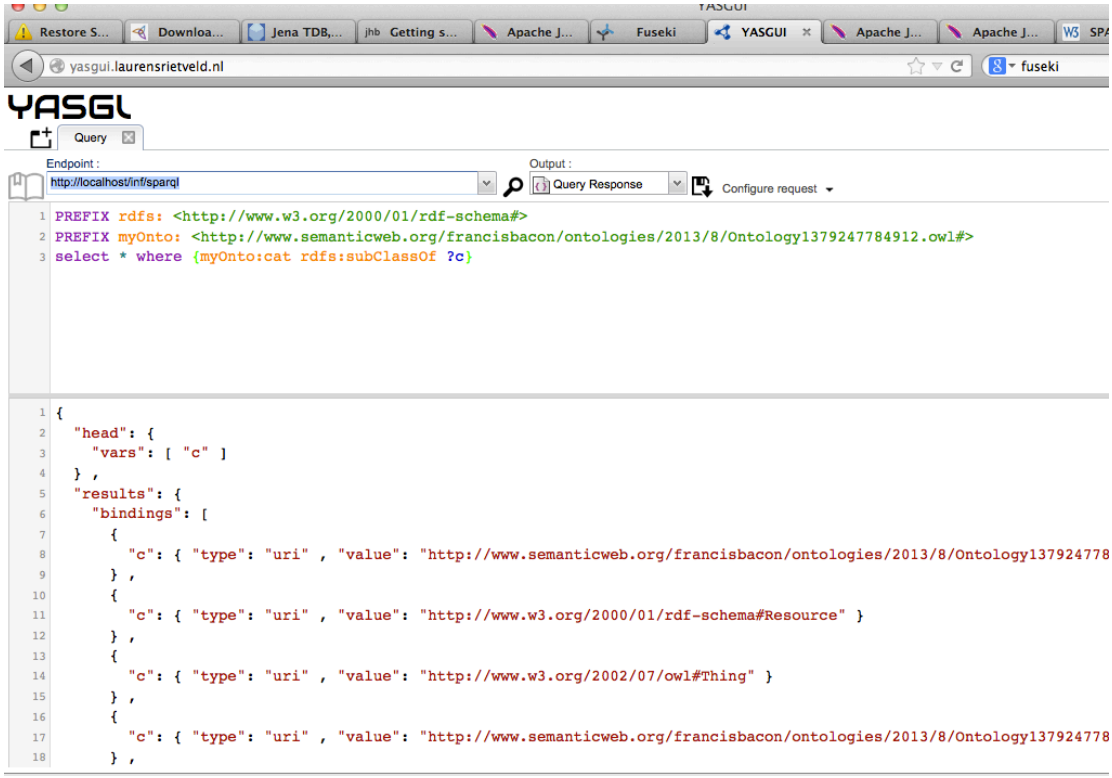


which results into:

(as you can see, the cat is an animal, thus Jena's rdfs reasoner did the job!)

Step 4: Use YASGUI to query your local endpoint

- goto http://yasgui.laurensrietveld.nl/
- add your local endpoint : http://localhost/inf/sparql
- Type your query (e.g. same as above)



--

Comments &/II Questions? Email: ronny@cs.vu.nl