GETTING STARTED WITH ONTOTEXT AND JENA

HENRIETTE HARMSE

In my previous post I explained how you can create an GraphDB repository and how you can update and query your repository using RDF4J. In this post I provide an example of how you can update and query a GraphDB repository using Jena. However, even thought the code works, there are some pitfalls.

1. A Quick Example

First you need to add Jena as a Maven dependency:

```
<dependency>
    <groupId>org.apache.jena</groupId>
    <artifactId>apache-jena-libs</artifactId>
    <version>3.7.0</version>
    <type>pom</type>
</dependency>
```

The Java code is straightforward:

```
package org.graphdb.jena.tutorial;

import org.apache.jena.query.QueryExecution;
import org.apache.jena.query.QueryExecutionFactory;
import org.apache.jena.query.QuerySolution;
import org.apache.jena.query.ResultSet;
import org.apache.jena.update.UpdateExecutionFactory;
import org.apache.jena.update.UpdateFactory;
import org.apache.jena.update.UpdateProcessor;
import org.apache.jena.update.UpdateRequest;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.slf4j.Marker;
import org.slf4j.MarkerFactory;

public class SimpleInsertQueryExample {
  private static Logger logger = LoggerFactory.getLogger(SimpleInsertQueryExample.class);
  // Why This Failure marker
  private static final Marker WTF_MARKER = MarkerFactory.getMarker("WTF");

  // GraphDB
  private static final String PERSONDATA_REPO_QUERY =
      "http://localhost:7200/repositories/PersonData";
  private static final String PERSONDATA_REPO_UPDATE =
      "http://localhost:7200/repositories/PersonData/statements";
```

Date: 29th June 2018.

```
private static String strInsert;
private static String strQuery;

static {
  strInsert =
      "INSERT DATA {"
       + "<http://dbpedia.org/resource/Grace_Hopper> "
       + "<http://dbpedia.org/ontology/birthDate>"
       + " \"1906-12-09\"^^<http://www.w3.org/2001/XMLSchema#date> ."
       + "<http://dbpedia.org/resource/Grace_Hopper> "
       + "<http://dbpedia.org/ontology/birthPlace> "
       + "<http://dbpedia.org/resource/New_York_City> ."
       + "<http://dbpedia.org/resource/Grace_Hopper> "
       + "<http://dbpedia.org/ontology/deathDate>"
       + " \"1992-01-01\"^^<http://www.w3.org/2001/XMLSchema#date> ."
       + "<http://dbpedia.org/resource/Grace_Hopper> "
       + "<http://dbpedia.org/ontology/deathPlace> "
       + "<http://dbpedia.org/resource/Arlington_County,_Virginia> ."
       + "<http://dbpedia.org/resource/Grace_Hopper> "
       + "<http://purl.org/dc/terms/description>"
       + " \"American computer scientist and United States Navy officer.\" ."
       + "<http://dbpedia.org/resource/Grace_Hopper> "
       + "<http://www.w3.org/1999/02/22-rdf-syntax-ns#type> "
       + "<http://dbpedia.org/ontology/Person> ."
       + "<http://dbpedia.org/resource/Grace_Hopper> "
       + "<http://xmlns.com/foaf/0.1/gender> \"female\" ."
       + "<http://dbpedia.org/resource/Grace_Hopper> "
       + "<http://xmlns.com/foaf/0.1/givenName> \"Grace\" ."
       + "<http://dbpedia.org/resource/Grace_Hopper> "
       + "<http://xmlns.com/foaf/0.1/name> \"Grace Hopper\" ."
       + "<http://dbpedia.org/resource/Grace_Hopper>"
       + " <http://xmlns.com/foaf/0.1/surname> \"Hopper\" ."
       + "}";

  strQuery =
      "SELECT ?name WHERE {?s <http://xmlns.com/foaf/0.1/name> ?name .}";
}

private static void insert() {
  UpdateRequest updateRequest = UpdateFactory.create(strInsert);
  UpdateProcessor updateProcessor = UpdateExecutionFactory
      .createRemote(updateRequest,
      PERSONDATA_REPO_UPDATE);
  updateProcessor.execute();
}

private static void query() {
  QueryExecution queryExecution = QueryExecutionFactory
      .sparqlService(PERSONDATA_REPO_QUERY, strQuery);
  for (ResultSet results = queryExecution.execSelect(); results.hasNext();) {
    QuerySolution qs = results.next();
    String strName = qs.get("?name").toString();
    logger.trace("name = " + strName);
    }
    queryExecution.close();
  }


  public static void main(String[] args) {
    try {
      insert();
      query();
    } catch (Throwable t) {
      logger.error(WTF_MARKER, t.getMessage(), t);
    }
  }
}
```

2. Some Pitfalls

The example I provided will insert RDF data into GraphDB and query it successfully. However, the data is inserted into the repository in the absence of a transaction. The transaction API of Jena is based on a `Dataset`. Historically Ontotext provided a Jena adapter with which a Jena `Dataset` could be created. However, based on my question on Stack Overflow in this regard, the Jena adapter is no longer supported by Ontotext. Hence, currently it is not clear to me how to enable transactions when using Jena to access GraphDB. So, if you know how to address this, please be so kind as to leave a comment with your insight!

3. Conclusion

In this post I provided a quick example of how you can access GraphDB using Jena. However, this example does not support transactions, and therefore you may want to look at rather using RDF4J with GraphDB. You can find this code at github.