

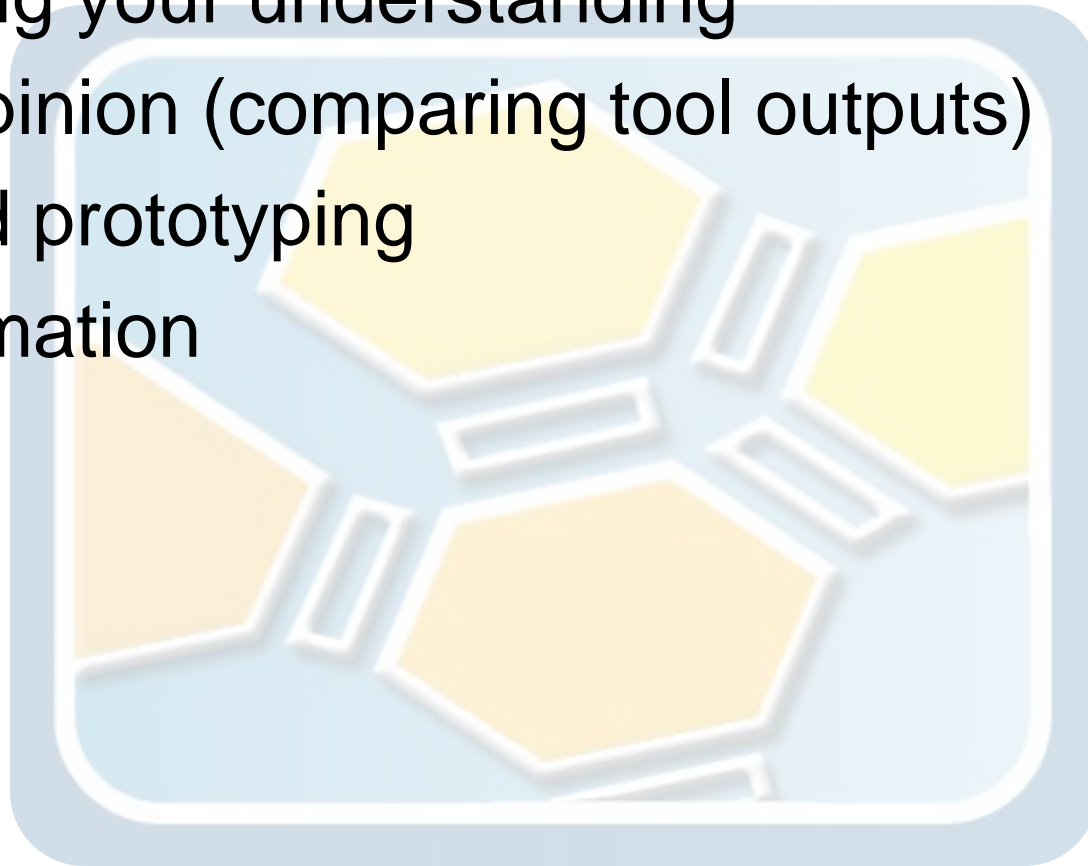
# Rapid Prototyping with the Jena Command Line Utilities

Mark Wallace  
Modus Operandi, Inc

# Motivation

2

- Dabbling in the semantic web
- Testing your understanding
- 2<sup>nd</sup> opinion (comparing tool outputs)
- Rapid prototyping
- Automation



# Set Up

3

- Download: <http://jena.sourceforge.net/>
- Install: Unpack to some directory/folder
- Set Java™ Classpath
  - ▣ I use a script (e.g. setjena.bat on windows)

```
@echo off
set CLASSPATH=
set CLASSPATH=%CLASSPATH%;\programs\Jena-2.6.4\lib\arq-2.8.7.jar
set CLASSPATH=%CLASSPATH%;\programs\Jena-2.6.4\lib\icu4j-3.4.4.jar
set CLASSPATH=%CLASSPATH%;\programs\Jena-2.6.4\lib\iri-0.8.jar
set CLASSPATH=%CLASSPATH%;\programs\Jena-2.6.4\lib\jena-2.6.4-tests.jar
set CLASSPATH=%CLASSPATH%;\programs\Jena-2.6.4\lib\jena-2.6.4.jar
set CLASSPATH=%CLASSPATH%;\programs\Jena-2.6.4\lib\junit-4.5.jar
set CLASSPATH=%CLASSPATH%;\programs\Jena-2.6.4\lib\log4j-1.2.13.jar
set CLASSPATH=%CLASSPATH%;\programs\Jena-2.6.4\lib\lucene-core-2.3.1.jar
set CLASSPATH=%CLASSPATH%;\programs\Jena-2.6.4\lib\slf4j-api-1.5.8.jar
set CLASSPATH=%CLASSPATH%;\programs\Jena-2.6.4\lib\slf4j-log4j12-1.5.8.jar
set CLASSPATH=%CLASSPATH%;\programs\Jena-2.6.4\lib\stax-api-1.0.1.jar
set CLASSPATH=%CLASSPATH%;\programs\Jena-2.6.4\lib\wstx-asl-3.2.9.jar
set CLASSPATH=%CLASSPATH%;\programs\Jena-2.6.4\lib\xercesImpl-2.7.1.jar
```

# General Tool Notes

4

- Standard file suffixes recognized by Jena
  - ▣ .owl, .rdf => RDF/XML (default; e.g. stdin, URL)
  - ▣ .nt .ntriples => N-Triples
  - ▣ .ttl, .n3 => Turtle or Notation 3 (N3)
- Help with tool usage:
  - ▣ use `-help` option, e.g.
    - `java jena.rdfcat -help`
- Can manipulate files or URIs
  - ▣ `java jena.rdfcat univ-bench.owl`
  - ▣ `java jena.rdfcat \`  
`http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl`

# Usage: Converting File Formats

5

- Jena `rdfcats` tool
- Convert to Turtle<sup>[1]</sup>
  - ▣ `java jena.rdfcats -out ttl univ-bench.owl`
- Convert to N-Triples<sup>[2]</sup>
  - ▣ `java jena.rdfcats -out ntriple \`  
`wgs84_pos.rdf`
  - ▣ `java jena.rdfcats -out ntriple \`  
`wgs84_pos.rdf > tmp.nt && notepad tmp.nt`
- Convert to RDF/XML<sup>[3]</sup>
  - ▣ `java jena.rdfcats veh.ttl`
  - ▣ `java jena.rdfcats -out xml veh.ttl`

# Usage: Merging RDF Files

6

- Jena **rdffcat** tool
  - Specify multiple files on command line
  - `java jena.rdffcat abox.ttl tbox.ttl`
  - Duplicates are removed in Jena merges
- Following `rdfs:seeAlso` and `owl:imports`<sup>[4]</sup>
  - Need to create local file: `location-mapping.n3`

```
@prefix lm: <http://jena.hpl.hp.com/2004/08/location-mapping#> .  
  
[] lm:mapping  
  [ lm:name "http://example.com/tbox"  
    ; lm:altName "file:tbox.owl" ] .
```

- `java jena.rdffcat -include abox.owl`

# Usage: Filtering Triples

7

## □ Using findstr or grep

- ▣ `java jena.rdfcat -out ntriple abox.ttl | findstr /v #type`
- ▣ `java jena.rdfcat -out ntriple abox.ttl | grep -v "#type"`

## □ Using sed<sup>[5]</sup> makes triples easier to read

- ▣ I use a script (filt.bat on Windows)

```
@echo off
sed "s$http://example.org/$$g;s$http://www.w3.org/1999/02/22-rdf-syntax-ns#$rdf:
$g;s$http://www.w3.org/2000/01/rdf-schema#$rdfs:$g;s$http://www.w3.org/2002/07/o
wl#$owl:$g"
```

- ▣ `java jena.rdfcat -out ntriple abox.ttl | filt`
- ▣ `java jena.rdfcat -out ntriple abox.ttl | filt | findstr /v rdf:type`

# Usage: Filtering Triples

8

```
C:\ SemTech 2011
C> java jena.rdfcat -out ntriple abox.ttl
<http://example.org/ex#Civic_8717383> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://example.org/ex#Civic> .
<http://example.org/ex#Civic_8717383> <http://example.org/ex#hasVin> "1H0243098430987" .
<http://example.org/ex#Truck_8sk4849> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://example.org/ex#Truck> .
<http://example.org/ex#Truck_8sk4849> <http://example.org/ex#hasVin> "FT948473AFDF829" .
<http://example.org/ex#Civic_9384953> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://example.org/ex#Civic> .
<http://example.org/ex#Civic_9384953> <http://example.org/ex#hasVin> "1H9384759384757" .

C> java jena.rdfcat -out ntriple abox.ttl | filt
<ex#Civic_8717383> <rdf:type> <ex#Civic> .
<ex#Civic_8717383> <ex#hasVin> "1H0243098430987" .
<ex#Truck_8sk4849> <rdf:type> <ex#Truck> .
<ex#Truck_8sk4849> <ex#hasVin> "FT948473AFDF829" .
<ex#Civic_9384953> <rdf:type> <ex#Civic> .
<ex#Civic_9384953> <ex#hasVin> "1H9384759384757" .

C> java jena.rdfcat -out ntriple abox.ttl | filt | findstr /v rdf:type
<ex#Civic_8717383> <ex#hasVin> "1H0243098430987" .
<ex#Truck_8sk4849> <ex#hasVin> "FT948473AFDF829" .
<ex#Civic_9384953> <ex#hasVin> "1H9384759384757" .

C>
```

# Usage: SPARQL Query

9

- Jena `sparql` tool
- SPARQL query of file
  - `java jena.sparql --data wgs84_pos.rdf "select * where {?s a ?o}"`
  - `java jena.sparql "select * from <file:wgs84_pos.rdf> where {?s a ?o}"`
- SPARQL query of URI
  - `java jena.sparql --data http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl "select * where {?s a ?o} limit 2"`
  - `java jena.sparql "select * from <http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl> where {?s a ?o} limit 2"`

# Usage: SPARQL Query

10

- SPARQL query of combined files
  - ▣ `java jena.sparql --data wgs84_pos.rdf --data univ-bench.owl "select * where {?s a ?o}"`
  - ▣ `java jena.sparql "select * from <file:wgs84_pos.rdf> from <file:univ-bench.owl> where {?s a ?o}"`
- SPARQL query with XML results
  - ▣ `java jena.sparql --results xml --data wgs84_pos.rdf "select * where {?s a ?o} limit 2"`

# Usage: SPARQL Query

11

- SPARQL query with query in a file
  - `java jena.sparql --data wgs84_pos.rdf`  
`--query q6.rq`



# Usage: Less keystrokes!

12

## □ Windows:

```
C> type rdfcat.bat
@echo off
java jena.rdfcat %*
```

```
C> type ntrip.bat
@echo off
java jena.rdfcat -out ntriple %* | sort
```

```
C> type ttl.bat
@echo off
java jena.rdfcat -out ttl %*
```

```
C> type sp.bat
@echo off
java jena.sparql %*
```

# Usage: Less keystrokes!

13

## □ Unix:

```
$ cat rdfcat
java jena.rdfcat $*

$ cat ntrip
java jena.rdfcat -out ntriple $* | sort

$ cat ttl
java jena.rdfcat -out ttl $*

$ cat sp
java jena.sparql $*
```

# Usage: Inference

14

- Jena comes with reasoners<sup>[6]</sup>
  - RDFS
  - OWL micro
  - OWL mini
  - OWL
  - a few others...
- However, Jena doesn't include an *inference command line utility*

# Coding an Infer Utility

15

```
import com.hp.hpl.jena.rdf.model.*;
import com.hp.hpl.jena.reasoner.*;
import com.hp.hpl.jena.vocabulary.*;

/** read RDF XML from standard in and write inferred model to standard out */
class infer {
    public static void main (String args[]) {

        // Create an empty model.
        Model model = ModelFactory.createDefaultModel();

        // Read the RDF/XML on standard in.
        model.read(System.in, "");

        // Create a reasoner.
        Reasoner reasoner = ReasonerRegistry.getRDFSReasoner();
        reasoner.setParameter(ReasonerVocabulary.PROPsetRDFSLevel,
            ReasonerVocabulary.RDFS_SIMPLE);

        // Create an inferred model.
        InfModel inf = ModelFactory.createInfModel(reasoner, model);

        // Write it to standard out.
        inf.write(System.out);
    }
}
```

# Running Inference

16

- Remember: reads from stdin
  - ▣ `java infer < wgs84_pos.rdf`
  - ▣ `type wgs84_pos.rdf | java infer`
- Since reading from stdin, convert to RDF/XML (can't guess format by file extension)
  - ▣ `rdffcat veh.ttl | java infer`
- Capturing output
  - ▣ `rdffcat veh.ttl | java infer > inf.rdf`
- Converting output
  - ▣ `rdffcat veh.ttl | java infer | ttl -x -`

# Usage: SPARQL Query with Inference

17

- SPARQL only queries info in the model
  - ▣ no inference implied in SPARQL itself [7]
  - ▣ so...
- Run inference first
- Then issue query against inferred results
- `java jena.rdfcat veh.ttl | java infer > veh-inf.rdf`
- `java jena.sparql --data veh-inf.rdf "select * where {?v a <http://example.org/ex#Vehicle>}"`

# Further Ideas

18

- "Diff" Tool (differences between RDF files)
  - ▣ Convert both files to sorted N-Triples, then
  - ▣ Run `diff` or `fc`
- Analyze Triples in Excel
  - ▣ Convert to N-Triples
  - ▣ Use `sed` tool to convert to CSV
  - ▣ Use MS Excel (or other) to sort, filter columns

# Coding a Pellet Infer Tool (Optional)

19

## □ A Pellet<sup>[8]</sup> reasoner exists for Jena

```
import com.hp.hpl.jena.rdf.model.*;
import com.hp.hpl.jena.reasoner.*;
import org.mindswap.pellet.jena.PelletReasonerFactory;

/** read RDF XML from standard in and write inferred model to standard out */
class pellinf {

    public static void main (String args[]) {
        // Create an empty model.
        Model model = ModelFactory.createDefaultModel();

        // Read the RDF/XML on standard in.
        model.read(System.in, "");

        // Create a reasoner.
        Reasoner reasoner = PelletReasonerFactory.theInstance().create();

        // Create an inferred model.
        InfModel inf = ModelFactory.createInfModel(reasoner, model);

        // Write it to standard out.
        inf.write(System.out);
    }
}
```

# References

20

- [1] Turtle: <http://www.w3.org/TeamSubmission/turtle/>
- [2] N-Triples: <http://www.w3.org/2001/sw/RDFCore/ntriples/>
- [3] RDF/XML Syntax Spec: <http://www.w3.org/TR/rdf-syntax-grammar/>
- [4] Jena File Manager: <http://jena.sourceforge.net/how-to/filemanager.html>
- [5] Windows Sed: <http://www.pement.org/sed/>
- [6] Jena Reasoners: <http://jena.sourceforge.net/inference/>
- [7] SPARQL Tutorial: <http://jena.sourceforge.net/ARQ/Tutorial/index.html>
- [8] Pellet reasoner: <http://clarkparsia.com/pellet/>
  
- Blog: <http://semapps.blogspot.com>