OntoSpy Documentation

Release 1.6.7

Michele Pasin

November 24, 2016

Contents

OntoSpy is a lightweight Python library and command line tool for inspecting and visualizing vocabularies encoded in the RDF family of languages.

See also:

- CheeseShop: https://pypi.python.org/pypi/ontospy
- Github: https://github.com/lambdamusic/ontospy
- Homepage: http://www.michelepasin.org/projects/ontospy
- Docs: http://ontospy.readthedocs.org

CHAPTER 1

In a nutshell

OntoSpy can be used either as an interactive command line interface (a repl) or as a Python package.

Calling the `ontospy` command from a terminal window launches a utility for scanning a knowledge model encoded in RDF (or any of its dialects e.g. RDFS, OWL or SKOS). For example, if you pass a valid graph URI (e.g. `ontospy http://purl.org/spar/frbr`) then OntoSpy will attempt to extract and print out any ontology-related information contained in that graph.

Many other options are available, in particular OntoSpy allows to load/save ontologies from/to a local repository so that they can be cached and quickly reloaded for inspection later on. All without leaving your terminal window!

Is OntoSpy for me?

Here are some reasons why you should use it:

- You have a bunch of RDF vocabularies you regularly need to interrogate, but do not want to load a full-blown ontology editor like Protege.
- You need to quickly generate documentation for an ontology, either as simple html pages or via some more elaborate interactive visualization.
- You love the command line and would never leave it no matter what.
- You are developing a Python application that needs to extract schema information from an RDF, SKOS or OWL vocabulary.

Note: OntoSpy does not offer any ontology-editing features, nor it can be used to interrogate a triplestore.

Quick example

Here's how it works from the command line (video link):

If used as a Python package, the basic workflow is the following: load a graph by instantiating the `Ontospy` class with a file containing RDFS, OWL or SKOS definitions; you get back an object that lets you interrogate the ontology. That's all!

Let's take a look at the Friend Of A Friend vocabulary.

```
In [1]: import ontospy
INFO:rdflib:RDFLib Version: 4.2.0

In [2]: model = ontospy.Ontospy("http://xmlns.com/foaf/0.1/")
----------
Loaded 631 triples from <http://xmlns.com/foaf/0.1/>
started scanning...
----------
Ontologies found...: 1
Classes found......: 14
Properties found...: 67
Annotation.........: 7
Datatype...........: 26
Object.............: 34
SKOS Concepts......: 0
----------

In [3]: model.classes
Out[3]:
[<Class *http://www.w3.org/2003/01/geo/wgs84_pos#SpatialThing*>,
 <Class *http://xmlns.com/foaf/0.1/Agent*>,
 <Class *http://xmlns.com/foaf/0.1/Document*>,
 <Class *http://xmlns.com/foaf/0.1/Group*>,
 <Class *http://xmlns.com/foaf/0.1/Image*>,
 <Class *http://xmlns.com/foaf/0.1/LabelProperty*>,
 <Class *http://xmlns.com/foaf/0.1/OnlineAccount*>,
 <Class *http://xmlns.com/foaf/0.1/OnlineChatAccount*>,
 <Class *http://xmlns.com/foaf/0.1/OnlineEcommerceAccount*>,
 <Class *http://xmlns.com/foaf/0.1/OnlineGamingAccount*>,
 <Class *http://xmlns.com/foaf/0.1/Organization*>,
 <Class *http://xmlns.com/foaf/0.1/Person*>,
 <Class *http://xmlns.com/foaf/0.1/PersonalProfileDocument*>,
 <Class *http://xmlns.com/foaf/0.1/Project*>]

In [4]: model.properties
```

```
Out[4]:
[<Property *http://xmlns.com/foaf/0.1/account*>,
 <Property *http://xmlns.com/foaf/0.1/accountName*>,
 <Property *http://xmlns.com/foaf/0.1/accountServiceHomepage*>,
 <Property *http://xmlns.com/foaf/0.1/age*>,
 <Property *http://xmlns.com/foaf/0.1/aimChatID*>,
 <Property *http://xmlns.com/foaf/0.1/based_near*>,
 <Property *http://xmlns.com/foaf/0.1/birthday*>,
 <Property *http://xmlns.com/foaf/0.1/currentProject*>,
 <Property *http://xmlns.com/foaf/0.1/depiction*>,
        ### etc....
        ]

In [5]: model.printClassTree()
[1]    http://www.w3.org/2003/01/geo/wgs84_pos#SpatialThing
[12]   ----_file_:Person
[2]    _file_:Agent
[4]    ----_file_:Group
[11]   ----_file_:Organization
[12]   ----_file_:Person
[3]    _file_:Document
[5]    ----_file_:Image
[13]   ----_file_:PersonalProfileDocument
[6]    _file_:LabelProperty
[7]    _file_:OnlineAccount
[8]    ----_file_:OnlineChatAccount
[9]    ----_file_:OnlineEcommerceAccount
[10]   ----_file_:OnlineGamingAccount
[14]   _file_:Project


In [6]: model.toplayer
Out[6]:
[<Class *http://www.w3.org/2003/01/geo/wgs84_pos#SpatialThing*>,
 <Class *http://xmlns.com/foaf/0.1/Agent*>,
 <Class *http://xmlns.com/foaf/0.1/Document*>,
 <Class *http://xmlns.com/foaf/0.1/LabelProperty*>,
 <Class *http://xmlns.com/foaf/0.1/OnlineAccount*>,
 <Class *http://xmlns.com/foaf/0.1/Project*>]

In [7]: model.getClass("document")
Out[7]:
[<Class *http://xmlns.com/foaf/0.1/Document*>,
 <Class *http://xmlns.com/foaf/0.1/PersonalProfileDocument*>]

In [8]: a_class = _[0]

In [9]: print(a_class.serialize())
@prefix ns1: <http://www.w3.org/2002/07/owl#> .
@prefix ns2: <http://www.w3.org/2003/06/sw-vocab-status/ns#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xml: <http://www.w3.org/XML/1998/namespace> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

<http://xmlns.com/foaf/0.1/Document> a rdfs:Class,
        ns1:Class ;
    rdfs:label "Document" ;
    rdfs:comment "A document." ;
    rdfs:isDefinedBy <http://xmlns.com/foaf/0.1/> ;
    ns1:disjointWith <http://xmlns.com/foaf/0.1/Organization>,
        <http://xmlns.com/foaf/0.1/Project> ;
    ns1:equivalentClass <http://schema.org/CreativeWork> ;
    ns2:term_status "stable" .



In [10]: a_class.parents()
Out[10]: []

In [11]: a_class.children()
Out[11]:
[<Class *http://xmlns.com/foaf/0.1/Image*>,
 <Class *http://xmlns.com/foaf/0.1/PersonalProfileDocument*>]
```

CHAPTER 3

Contents

3.1 Installation

3.1.1 Prerequisites

- Python (tested on 2.x or 3.x)
- A python package manager: setuptools or pip.

3.1.2 Installation:

Once you have a package manager installed, get OntoSpy from the Python Package Index:

`easy_install ontospy` or `pip install ontospy`

The python library, its dependencies and all of its command-line executables will be installed.

Note: if you're upgrading from an older version, make sure you use the -U flag: `pip install ontospy -U`

3.1.3 Credits

OntoSpy couldn't exist without the following libraries.

- rdflib
- colorama
- readline
- django
- requests
- click
- pygments
- pyfiglet

Note: all of the above will install automatically when you install OntoSpy.

3.2 Quick Start

A few examples showing how Ontospy can be used in Python programs, or interactively using the Python console.

Warning: This documentation is still in draft mode.

Loading and querying the Friend Of A Friend vocabulary:

```
In [1]: import ontospy
INFO:rdflib:RDFLib Version: 4.2.0

In [2]: model = ontospy.Ontospy("http://xmlns.com/foaf/0.1/")
----------
Loaded 631 triples from <http://xmlns.com/foaf/0.1/>
started scanning...
----------
Ontologies found...: 1
Classes found......: 14
Properties found...: 67
Annotation.........: 7
Datatype...........: 26
Object.............: 34
SKOS Concepts......: 0
----------

In [3]: model.classes
Out[3]:
[<Class *http://www.w3.org/2003/01/geo/wgs84_pos#SpatialThing*>,
 <Class *http://xmlns.com/foaf/0.1/Agent*>,
 <Class *http://xmlns.com/foaf/0.1/Document*>,
 <Class *http://xmlns.com/foaf/0.1/Group*>,
 <Class *http://xmlns.com/foaf/0.1/Image*>,
 <Class *http://xmlns.com/foaf/0.1/LabelProperty*>,
 <Class *http://xmlns.com/foaf/0.1/OnlineAccount*>,
 <Class *http://xmlns.com/foaf/0.1/OnlineChatAccount*>,
 <Class *http://xmlns.com/foaf/0.1/OnlineEcommerceAccount*>,
 <Class *http://xmlns.com/foaf/0.1/OnlineGamingAccount*>,
 <Class *http://xmlns.com/foaf/0.1/Organization*>,
 <Class *http://xmlns.com/foaf/0.1/Person*>,
 <Class *http://xmlns.com/foaf/0.1/PersonalProfileDocument*>,
 <Class *http://xmlns.com/foaf/0.1/Project*>]

In [4]: model.properties
Out[4]:
[<Property *http://xmlns.com/foaf/0.1/account*>,
 <Property *http://xmlns.com/foaf/0.1/accountName*>,
 <Property *http://xmlns.com/foaf/0.1/accountServiceHomepage*>,
 <Property *http://xmlns.com/foaf/0.1/age*>,
 <Property *http://xmlns.com/foaf/0.1/aimChatID*>,
 <Property *http://xmlns.com/foaf/0.1/based_near*>,
 <Property *http://xmlns.com/foaf/0.1/birthday*>,
 <Property *http://xmlns.com/foaf/0.1/currentProject*>,
 <Property *http://xmlns.com/foaf/0.1/depiction*>,
 <Property *http://xmlns.com/foaf/0.1/depicts*>,
 <Property *http://xmlns.com/foaf/0.1/dnaChecksum*>,
 <Property *http://xmlns.com/foaf/0.1/familyName*>,
 <Property *http://xmlns.com/foaf/0.1/family_name*>,
 <Property *http://xmlns.com/foaf/0.1/firstName*>,
 <Property *http://xmlns.com/foaf/0.1/focus*>,
 <Property *http://xmlns.com/foaf/0.1/fundedBy*>,
 <Property *http://xmlns.com/foaf/0.1/geekcode*>,
 <Property *http://xmlns.com/foaf/0.1/gender*>,
 <Property *http://xmlns.com/foaf/0.1/givenName*>,
 <Property *http://xmlns.com/foaf/0.1/holdsAccount*>,
 <Property *http://xmlns.com/foaf/0.1/homepage*>,
 <Property *http://xmlns.com/foaf/0.1/icqChatID*>,
 <Property *http://xmlns.com/foaf/0.1/img*>,
 <Property *http://xmlns.com/foaf/0.1/interest*>,
 <Property *http://xmlns.com/foaf/0.1/isPrimaryTopicOf*>,
 <Property *http://xmlns.com/foaf/0.1/jabberID*>,
 <Property *http://xmlns.com/foaf/0.1/knows*>,
 <Property *http://xmlns.com/foaf/0.1/lastName*>,
 <Property *http://xmlns.com/foaf/0.1/logo*>,
 <Property *http://xmlns.com/foaf/0.1/made*>,
 <Property *http://xmlns.com/foaf/0.1/maker*>,
 <Property *http://xmlns.com/foaf/0.1/mbox*>,
 <Property *http://xmlns.com/foaf/0.1/mbox_sha1sum*>,
 <Property *http://xmlns.com/foaf/0.1/member*>,
 <Property *http://xmlns.com/foaf/0.1/membershipClass*>,
 <Property *http://xmlns.com/foaf/0.1/msnChatID*>,
 <Property *http://xmlns.com/foaf/0.1/myersBriggs*>,
 <Property *http://xmlns.com/foaf/0.1/name*>,
 <Property *http://xmlns.com/foaf/0.1/nick*>,
 <Property *http://xmlns.com/foaf/0.1/openid*>,
 <Property *http://xmlns.com/foaf/0.1/page*>,
 <Property *http://xmlns.com/foaf/0.1/pastProject*>,
 <Property *http://xmlns.com/foaf/0.1/phone*>,
 <Property *http://xmlns.com/foaf/0.1/plan*>,
 <Property *http://xmlns.com/foaf/0.1/primaryTopic*>,
 <Property *http://xmlns.com/foaf/0.1/publications*>,
 <Property *http://xmlns.com/foaf/0.1/schoolHomepage*>,
 <Property *http://xmlns.com/foaf/0.1/sha1*>,
 <Property *http://xmlns.com/foaf/0.1/skypeID*>,
 <Property *http://xmlns.com/foaf/0.1/status*>,
 <Property *http://xmlns.com/foaf/0.1/surname*>,
 <Property *http://xmlns.com/foaf/0.1/theme*>,
 <Property *http://xmlns.com/foaf/0.1/thumbnail*>,
 <Property *http://xmlns.com/foaf/0.1/tipjar*>,
 <Property *http://xmlns.com/foaf/0.1/title*>,
 <Property *http://xmlns.com/foaf/0.1/topic*>,
 <Property *http://xmlns.com/foaf/0.1/topic_interest*>,
 <Property *http://xmlns.com/foaf/0.1/weblog*>,
 <Property *http://xmlns.com/foaf/0.1/workInfoHomepage*>,
 <Property *http://xmlns.com/foaf/0.1/workplaceHomepage*>,
 <Property *http://xmlns.com/foaf/0.1/yahooChatID*>,
 <Property *http://purl.org/dc/elements/1.1/date*>,
 <Property *http://purl.org/dc/elements/1.1/description*>,
 <Property *http://purl.org/dc/elements/1.1/title*>,
 <Property *http://www.w3.org/2003/06/sw-vocab-status/ns#term_status*>,
 <Property *http://xmlns.com/wot/0.1/assurance*>,
 <Property *http://xmlns.com/wot/0.1/src_assurance*>]

In [5]: model.printClassTree()
[1]    http://www.w3.org/2003/01/geo/wgs84_pos#SpatialThing
[12]   ----_file_:Person
```

```
[2]     _file_:Agent
[4]     ----_file_:Group
[11]    ----_file_:Organization
[12]    ----_file_:Person
[3]     _file_:Document
[5]     ----_file_:Image
[13]    ----_file_:PersonalProfileDocument
[6]     _file_:LabelProperty
[7]     _file_:OnlineAccount
[8]     ----_file_:OnlineChatAccount
[9]     ----_file_:OnlineEcommerceAccount
[10]    ----_file_:OnlineGamingAccount
[14]    _file_:Project


In [6]: model.toplayer
Out[6]:
[<Class *http://www.w3.org/2003/01/geo/wgs84_pos#SpatialThing*>,
 <Class *http://xmlns.com/foaf/0.1/Agent*>,
 <Class *http://xmlns.com/foaf/0.1/Document*>,
 <Class *http://xmlns.com/foaf/0.1/LabelProperty*>,
 <Class *http://xmlns.com/foaf/0.1/OnlineAccount*>,
 <Class *http://xmlns.com/foaf/0.1/Project*>]

In [7]: model.getClass("document")
Out[7]:
[<Class *http://xmlns.com/foaf/0.1/Document*>,
 <Class *http://xmlns.com/foaf/0.1/PersonalProfileDocument*>]

In [8]: d = _[0]

In [9]: print(d.serialize())
@prefix ns1: <http://www.w3.org/2002/07/owl#> .
@prefix ns2: <http://www.w3.org/2003/06/sw-vocab-status/ns#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xml: <http://www.w3.org/XML/1998/namespace> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

<http://xmlns.com/foaf/0.1/Document> a rdfs:Class,
        ns1:Class ;
    rdfs:label "Document" ;
    rdfs:comment "A document." ;
    rdfs:isDefinedBy <http://xmlns.com/foaf/0.1/> ;
    ns1:disjointWith <http://xmlns.com/foaf/0.1/Organization>,
        <http://xmlns.com/foaf/0.1/Project> ;
    ns1:equivalentClass <http://schema.org/CreativeWork> ;
    ns2:term_status "stable" .



In [10]: d.parents()
Out[10]: []

In [11]: d.children()
Out[11]:
[<Class *http://xmlns.com/foaf/0.1/Image*>,
 <Class *http://xmlns.com/foaf/0.1/PersonalProfileDocument*>]
```

Note: The Ontospy object can be instantiated by passing a single file location or a folder path. In the second case it'll try to load any RDF file found in that path (recursively). For example, the second option can be handy if you have a knowledge model which is split into multiple files.

3.3 Command Line Usage

This page shows how to use Ontospy from the command line.

These are the commands available:

- `ontospy`: used to launch the Ontospy parser.
- `ontospy-shell`: used to launch the Ontospy command line interface.
- `ontospy-viz`: used to the launch the Ontospy visualization library.

Note: If you install OntosPy via one of the suggested methods, appropriate executables for your platform should be compiled automatically and added to *usr/local/bin* (by default on unix-based systems).

3.3.1 The `ontospy` command

A good place to start is the -h option:

```
Usage: ontospy [OPTIONS] [SOURCES]...

  Ontospy is a command line inspector for RDF/OWL knowledge models.

  Examples:

  Inspect a local RDF file:

  > ontospy /path/to/mymodel.rdf

  List ontologies available in the local library:

  > ontospy -l

  Open FOAF vocabulary and save it to the local library:

  > ontospy http://xmlns.com/foaf/spec/ -s

  More info: <ontospy.readthedocs.org>

Options:
  -b, --bootstrap  BOOTSTRAP: bootstrap the local library.
  -c, --cache      CACHE: force caching of the local library (for faster
                                loading).
  -d, --delete     DELETE: remove an ontology from the local library.
  -l, --library    LIBRARY: list ontologies saved in the local library.
  -s, --save       SAVE: save a file/folder/url into the local library.
  -r, --reset      RESET: delete all files in the local library.
  -u, --update     UPDATE: enter new path for the local library.
  -v, --verbose    VERBOSE: show entities labels as well as URIs.
  -w, --web         WEB: import ontologies from remote repositories.
  -h, --help        Show this message and exit.
```

Just calling `ontospy` without any argument has the effect of listing out all RDF models saved in the local library. The first time you run it, there are none obviously so you might want to run the `ontospy -b` option to get started.

Alternatively, you can also pass a valid graph URI as an argument to the `ontospy` command in order to print out useful ontology information:

```
> ontospy http://www.ifomis.org/bfo/1.1

# prints info about BFO resolving redirects etc..

Ontospy v1.7
Local library: </Users/michele.pasin/Dropbox/ontologies/ontospy-library/>
----------
.. trying rdf serialization: <xml>
..... success!
----------
Loaded 429 triples from <https://raw.githubusercontent.com/BFO-ontology/BFO/releases/1.1.1/bfo.owl>
started scanning...
----------
Ontologies found...: 1
Classes found......: 39
Properties found...: 9
Annotation.........: 9
Datatype...........: 0
Object.............: 0
SKOS Concepts......: 0
----------

Ontology Annotations
-----------
http://www.ifomis.org/bfo/1.1
=> http://purl.org/dc/elements/1.1/source
.... Pierre Grenon: "BFO in a Nutshell: A Bi-categorial Axiomatization of BFO and Comparison with DOI
=> http://purl.org/dc/elements/1.1/language
.... en
=> http://purl.org/dc/elements/1.1/title
.... Basic Formal Ontology (BFO)
=> http://www.w3.org/1999/02/22-rdf-syntax-ns#type
.... http://www.w3.org/2002/07/owl#Ontology
### ...etc....



Class Taxonomy
----------
bfo:Entity
----snap:Continuant
--------snap:DependentContinuant
------------snap:GenericallyDependentContinuant
------------snap:SpecificallyDependentContinuant
----------------snap:Quality
----------------snap:RealizableEntity
--------------------snap:Disposition
--------------------snap:Function
--------------------snap:Role
--------snap:IndependentContinuant
```

```
------------snap:MaterialEntity
----------------snap:FiatObjectPart
----------------snap:Object
----------------snap:ObjectAggregate
------------snap:ObjectBoundary
------------snap:Site
--------snap:SpatialRegion
------------snap:OneDimensionalRegion
------------snap:ThreeDimensionalRegion
------------snap:TwoDimensionalRegion
------------snap:ZeroDimensionalRegion
----span:Occurrent
--------span:ProcessualEntity
------------span:FiatProcessPart
------------span:Process
------------span:ProcessAggregate
------------span:ProcessBoundary
------------span:ProcessualContext
--------span:SpatiotemporalRegion
------------span:ConnectedSpatiotemporalRegion
----------------span:SpatiotemporalInstant
----------------span:SpatiotemporalInterval
------------span:ScatteredSpatiotemporalRegion
--------span:TemporalRegion
------------span:ConnectedTemporalRegion
----------------span:TemporalInstant
----------------span:TemporalInterval
------------span:ScatteredTemporalRegion

Property Taxonomy
----------
dc:contributor
dc:creator
dc:format
dc:identifier
dc:language
dc:publisher
dc:rights
dc:source
dc:title

----------
Time:      3.42s
```

Note: You can pass the Ontospy command either a single file or a folder path. In the second case it'll try to load any RDF file found in that path (recursively). For example, the second option can be handy if you have a knowledge model which is split into multiple files.

3.3.2 The `ontospy-shell` command

```
> ontospy-shell -h
Usage: ontospy-shell [OPTIONS] [SOURCE]...

  This application launches the OntoSpy interactive shell.
  Note: if a local path or URI of an RDF model is provided, that gets loaded
  into the shell by default. E.g.:

  > ontospy-shell path/to/mymodel.rdf

Options:
  -h, --help  Show this message and exit.
```

Calling `ontospy-shell` without any argument launches the shell. The shell is an interactive environment that lets you import, load and inspect vocabularies. For more examples on how that works, take a look at this video:

Note: you can pass an argument in order to pre-load an RDF graph into the interactive session.

3.3.3 The `ontospy-viz` command

This utility allows to generate documentation for an RDF vocabulary, using visualization algorithms that create simple HTML pages, Markdown files, or more complex javascript interactive charts based on D3.js.

```
> ontospy-viz -h
Usage: ontospy-viz [OPTIONS] [SOURCE]...

  This application launches the OntoSpy visualization tool.

  Example:      > ontospy-viz path/to/mymodel.rdf

  Note: if the location of an RDF model is not provided, a selection can be
  made from the contents of the local ontospy library folder.

Options:
  -o, --outputpath TEXT  Output path (default: home folder)
  -h, --help             Show this message and exit.
```

Some of the export options are still quite experimental, so it's advisable to try out more than one if you're not happy with the results.

```
1) Javadoc
2) Markdown
3) Split Columns
4) Dendogram
5) Pack Hierarchy
6) Bubble Chart
7) Cluster Tree
8) Bar Hierarchy
9) Partition Table
10) Tree Pie
```

CHAPTER 4

Indices and tables

- genindex
- modindex
- search