deti universidade de aveiro
departamento de electrónica,
telecomunicações e informática

Semantic Web

Engenharia de Dados e Conhecimento

2019/2020

A Biblioteca RDFLib

Introdução

- A implementação anterior de uma *triplestore* forneceu-nos os conhecimentos sobre como funciona esse tipo de sistemas.

- Poderíamos agora implementar repositórios, *parsers* e *serializers* para os vários formatos de representação de RDF e criar a nossa própria biblioteca funcional para lidar com este tipo de dados.

- Contudo, tal levava-nos para fora do nosso principal objetivo: implementar aplicações semânticas para a web.

RDFLib

- A RDFLib consiste numa biblioteca leve, mas bastante completa, para aceder a estruturas RDF.

- É implementada em Python e por isso de usabilidade superior a partir desta linguagem.

RDFLib

- Instalação
  - Download do *package* a partir de:
    - <https://pypi.python.org/pypi/rdflib/4.2.2>
  - Ou uso do comando:
    - “pip install rdflib==4.2.2”

- Documentação
  - Download a partir de:
    - <https://rdflib.readthedocs.org>
  - Ou criar a documentação a partir do *package*:
    - Instalar o Sphinx: “pip install sphinx”
    - Executar o comando na pasta “docs” do package: “make”

RDFLib

- Leitura de um grafo

```
import rdflib
from rdflib import Graph
_graph = ConjunctiveGraph()
_graph.parse("persons.nt", format="nt")
```

RDFLib

- Saber os predicados do grafo

```
lista = set(_graph.predicates())
    for a in lista:
        print(a)
---
http://xmlns.com/foaf/0.1/givenname
http://xmlns.com/foaf/0.1/name
http://www.w3.org/1999/02/22-rdf-syntax-ns#type
http://xmlns.com/foaf/0.1/surname
```

RDFLib

- Listas de Sujeitos
  - dados, ou não, o predicado e o objeto

```
subjects(predicate=None, object=None)
```

- Listas de Objetos
  - Dados, ou não, o sujeito e o predicado

```
graph.objects(subject=None, predicate=None)
```

- Ver documentação
  - http://rdflib.readthedocs.io/en/stable/apidocs/rdflib.html#rdflib.graph.ConjunctiveGraph

RDFLib

- Filtragem de triplos (função triples)

```
lista = _graph.triples(
        (rdflib.URIRef('http://dbpedia.org/resource/14th_Dalai_Lama'),
         None, None))
    for a in lista:
        print(a)
---
(rdflib.URIRef('http://dbpedia.org/resource/14th_Dalai_Lama'),
  rdflib.URIRef('http://www.w3.org/1999/02/22-rdf-syntax-
  ns#type'), rdflib.URIRef('http://xmlns.com/foaf/0.1/Person'))
(rdflib.URIRef('http://dbpedia.org/resource/14th_Dalai_Lama'),
  rdflib.URIRef('http://xmlns.com/foaf/0.1/givenname'),
  rdflib.Literal(u'Tenzin', lang='de'))
(rdflib.URIRef('http://dbpedia.org/resource/14th_Dalai_Lama'),
  rdflib.URIRef('http://xmlns.com/foaf/0.1/name'),
  rdflib.Literal(u'Tenzin Gyatso', lang='de'))
...
```

RDFLib

- Representação noutros formatos
  - Para RDF/XML

```
of = open("persons.xml", "wb")
of.write(_graph.serialize(format="pretty-xml"))
of.close()
```

  - Para N3

```
of = open("persons.n3", "wb")
of.write(_graph.serialize(format="n3"))
of.close()
```

RDFLib

- RDF/XML

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF
  xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'
  xmlns:_3='http://xmlns.com/foaf/0.1/'
>
  <_3:Person rdf:about="http://dbpedia.org/resource/A.U._Fuimaono">
    <_3:name xml:lang="de">A.U. Fuimaono</_3:name>
    <_3:givenname xml:lang="de">A.U.</_3:givenname>
    <_3:surname xml:lang="de">Fuimaono</_3:surname>
  </_3:Person>
  <_3:Person rdf:about="http://dbpedia.org/resource/A._A._Milne">
    <_3:name xml:lang="de">A. A. Milne</_3:name>
    <_3:givenname xml:lang="de">A. A.</_3:givenname>
    <_3:surname xml:lang="de">Milne</_3:surname>
  </_3:Person>
```

RDFLib

- RDF/N3

```
@prefix _3: <http://xmlns.com/foaf/0.1/>.
@prefix _5: <http://dbpedia.org/resource/%27H%27.>.
@prefix _6: <http://dbpedia.org/resource/1980>.
@prefix _7: <http://dbpedia.org/resource/14>.
@prefix _8: <http://dbpedia.org/resource/%22Weird_Al%22>.
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.

 _8:_Yankovic a _3:Person;
     _3:givenname "\\\"Weird Al\\\""@de;
     _3:name "\\\"Weird Al\\\" Yankovic"@de;
     _3:surname "Yankovic"@de.

 _5:_Jones a _3:Person;
     _3:givenname "\\'H\\'"@de;
     _3:name "\\'H\\' Jones"@de;
     _3:surname "Jones"@de.
```

RDFLib

- Saber que *namespaces* existem

```
for prefix, uri in _graph.namespaces():
        print prefix, uri
```

- Criar *namespaces*

```
rdf = rdflib.Namespace(‘
http://www.w3.org/1999/02/22-rdf-syntax-ns#’)
```

RDFLib

- Inserir novos triplos

```
eu = rdflib.URIRef('http://www.ua.pt/htz@ua.pt')
fr = rdflib.URIRef('http://dbpedia.org/resource/14th_Dalai_Lama')
rdf = rdflib.Namespace('http://www.w3.org/1999/02/22-rdf-
  syntax-ns#')
foaf = rdflib.Namespace('http://xmlns.com/foaf/0.1/')
_graph.add((eu, rdf['type'], foaf['person']))
_graph.add((eu, foaf['friend'], fr))
_graph.add((fr, foaf['knows'], eu))
```

RDFLib

- Bases de Dados

```
g =  rdflib.Graph('SQLite')
g.open('./persons.db', create=True)
for t in _graph.triples((None,None,None)):
    g.add(t)
g.commit()
g.close()
```

- Necessita do *package* “rdflib-sqlite”
  - Comando: “pip install rdflib-sqlite”