ELECTRONIC PUBLISHING
AND DIGITAL STORYTELLING
Lesson 3

PRELIMINARIES OF SEMANTIC WEB

TABLE OF CONTENTS

RDF DATA MODEL

THE WEB OF DATA

WWW = URI + HTTP + HTML

URI = UNIFORM RESOURCE IDENTIFIER
HTTP = HYPERTEXT TRANSFER PROTOCOL
HTML = HYPERTEXT MARKUP LANGUAGE
In the WoD a URI is a ***persistent*** **conceptual mapping** to a real entity (e.g. a person) and not to a HTML page.

If the location of a HTML file describing the entity changes, the URI does not change.
Many HTML pages can describe the same entity identified by the same URI.
A HTML page can include information about many entities, hence being linked to many URIs.

THE WEB OF DATA

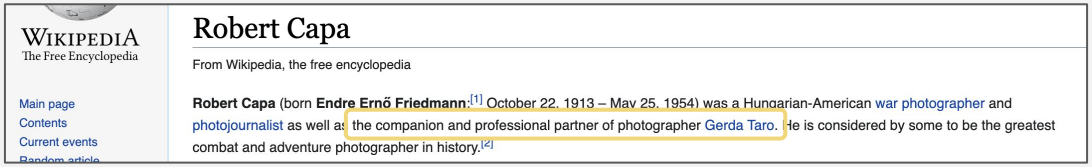
RDF = URI + URI + URI

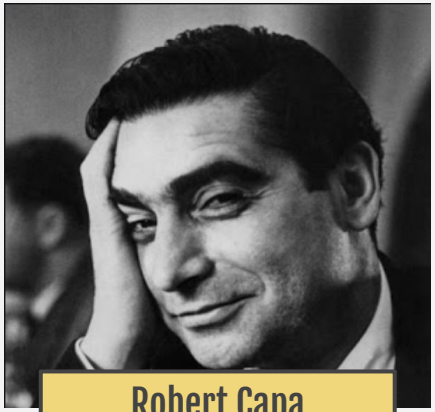
RDF = RESOURCE DESCRIPTION FRAMEWORK

In the WoD a URI identifies both real entities and the relations (links) between them.
Every piece of information is represented as **a triplet of URIs**, identifying respectively a subject, a predicate, and an object.

AN EXAMPLE

https://en.wikipedia.org/wiki/Robert_Capa





Robert Capa

Has spouse

Gerda Taro

http://dbpedia.org/resource/Robert_Capa

http://example.org/hasSpouse

http://dbpedia.org/resource/Gerda_Taro

AN EXAMPLE

<http://dbpedia.org/resource/Robert_Capa> <http://example.org/hasSpouse> <http://dbpedia.org/resource/Gerda_Taro> .

Store as -> **data.rdf**

ELECTRONIC PUBLISHING
AND DIGITAL STORYTELLING
Lesson 3

ORGANISING KNOWLEDGE

CLASSIFY WHAT IS DESCRIBED

Person

property

Person

An entity that can be classified as a person

A property characterising people, that relates people between each other.

An entity that can be classified as a person

IDENTIFY TERMS IN FORMAL VOCABULARIES

Person

property

Person

http://dbpedia.org/ontology/Person

http://dbpedia.org/ontology/Person

http://example.org/ontology/hasSpouse

IDENTIFY VOCABULARIES BY NAMESPACE

Person

property

Person

http://dbpedia.org/ontology/Person

http://dbpedia.org/ontology/Person

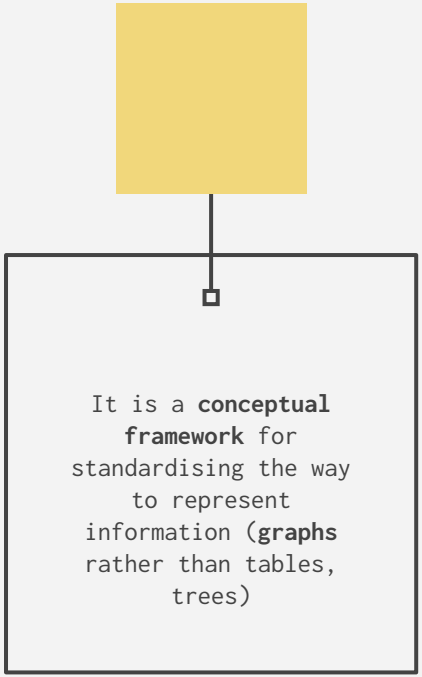
http://example.org/ontology/hasSpouse

AN EXAMPLE

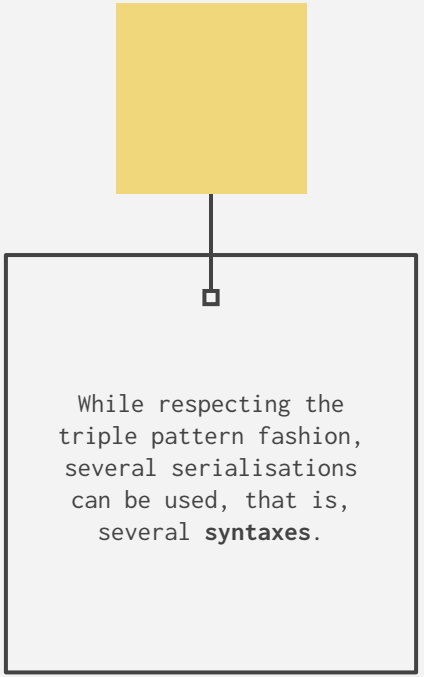
```
<http://dbpedia.org/ontology/Person> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.w3.org/2002/07/owl#Class> .
<http://dbpedia.org/ontology/Person> <http://www.w3.org/2000/01/rdf-schema#label> “Person” .
<http://dbpedia.org/ontology/Person> <http://www.w3.org/2000/01/rdf-schema#comment> “A human being” .
<http://example.org/ontology/hasSpouse> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
       <http://www.w3.org/2002/07/owl#ObjectProperty> .
<http://example.org/ontology/hasSpouse> <http://www.w3.org/2000/01/rdf-schema#label> “has spouse”.
```

Store as -> **ontology.rdf**

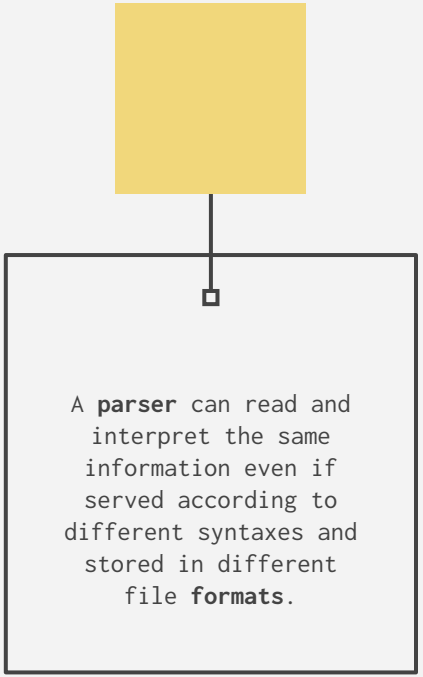
RDF IS NOT A FORMAT!



It is a **conceptual framework** for standardising the way to represent information (**graphs** rather than tables, trees)



While respecting the triple pattern fashion, several serialisations can be used, that is, several **syntaxes**.



A **parser** can read and interpret the same information even if served according to different syntaxes and stored in different file **formats**.

AN EXAMPLE

```
<http://dbpedia.org/resource/Robert_Capa> <http://example.org/hasSpouse> <http://dbpedia.org/resource/Gerda_Taro> .
```

data.nt

```
@prefix db: <http://dbpedia.org/resource/> .
@prefix ex: <http://example.org/> .

db:Robert_Capa ex:hasSpouse db:Gerda_Taro .
```

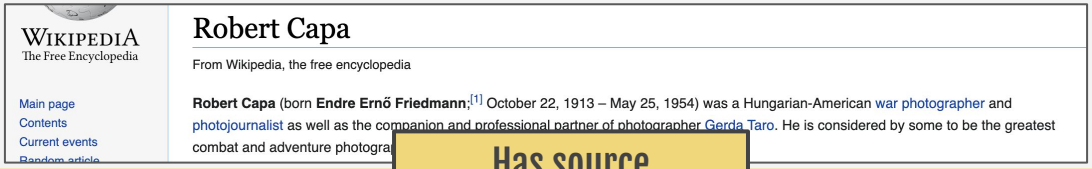
data.ttl

```
<?xml version="1.0" encoding="utf-8" ?>
<rdf:RDF xmlns:db="http://dbpedia.org/resource/" xmlns:ex="http://example.org/">
	<rdf:Description rdf:about="http://dbpedia.org/resource/Robert_Capa">
		<ex:hasSpouse rdf:resource="http://dbpedia.org/resource/Gerda_Taro" />
	</rdf:Description>
</rdf:RDF>
```

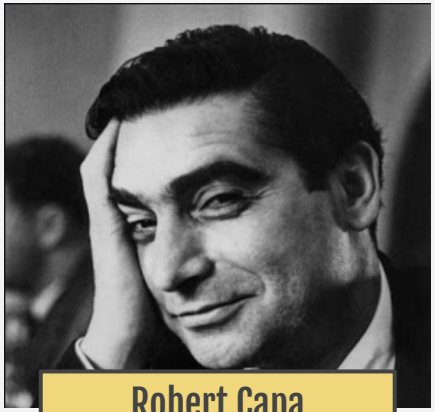
data.xml

WHEN TRIPLES ARE NOT ENOUGH

https://en.wikipedia.org/wiki/Robert_Capa



Has source



Robert Capa

Has spouse

Gerda Taro

http://dbpedia.org/resource/Robert_Capa

http://example.org/hasSpouse

http://dbpedia.org/resource/Gerda_Taro

WRAP TRIPLES IN NAMED GRAPHS

Robert Capa has spouse Gerda Taro

Has source

Wikipedia page

http://example.org/resource/Robert_Capa_hasSpouse_Gerda_Taro

http://example.org/hasSource

https://en.wikipedia.org/wiki/Robert_Capa

NAMED GRAPHS

```
<http://dbpedia.org/resource/Robert_Capa> <http://example.org/hasSpouse> <http://dbpedia.org/resource/Gerda_Taro>
<http://example.org/resource/Robert_Capa_hasSpouse_Gerda_Taro> .
<http://example.org/resource/Robert_Capa_hasSpouse_Gerda_Taro> <http://example.org/hasSource>
<https://en.wikipedia.org/wiki/Robert_Capa> <http://example.org/resource/Robert_Capa_hasSpouse_Gerda_Taro> .
```


data.nq

```
@prefix db: <http://dbpedia.org/resource/> .
@prefix ex: <http://example.org/> .
@prefix wiki: <https://en.wikipedia.org/wiki/> .

ex:Robert_Capa_hasSpouse_Gerda_Taro {
        db:Robert_Capa ex:hasSpouse db:Gerda_Taro .
        ex:Robert_Capa_hasSpouse_Gerda_Taro ex:hasSource wiki:Robert_Capa .
}
```

data.trig

CASE STUDY: ARTchives

MAIN ENTITIES IN ARTchives

A person that is responsible for creating one or more archival collections.

ART HISTORIANS

The materials produced by art historians during their activities. May include letters, photos, expertises.

COLLECTIONS

The cultural institutions preserving art historians' collections.

KEEPERS

RELATED ENTITIES

Artists and artistic movements studied by the art historian.
Other art historians.
Universities and research centres.

PEOPLE AND ORGANISATIONS

Types of materials.
Artists and movements.
Correspondants. Places.
Bibliography.

CONTENTS

Cities and contact addresses. Dates.
Descriptions (biographies, scope and content)

OTHER

DATA IN ARTchives

Federico Zeri	http://www.wikidata.org/entity/Q1089074
Coll. Warburg	https://w3id.org/artchives/collectionfondo-aby-warburg
main subject	http://www.wikidata.org/prop/direct/P921
Human	http://www.wikidata.org/entity/Q5

<https://w3id.org/artchives/collectionfondo-aby-warburg>
<http://www.wikidata.org/prop/direct/P921>
<http://www.wikidata.org/entity/Q1089074>
<https://w3id.org/artchives/1598630286-3009102/> .

Data are created by either reusing URIs available in **Wikidata** or by minting new URIs when no entity exist in Wikidata.

Likewise, classes and properties belong to the **Wikidata vocabulary** as much as possible.

For each collection, data about collection, historian and keeper are stored in a dedicated **named graphs**.

PROBLEMS IN ARTchives

User-generated data:
Duplication of data (e.g. labels).
Wrong reconciliation.
Misspelling.

Design choices:
Same URIs have different values in different Named Graphs (e.g. bibliographies).

Data about historians and keepers appear only in the graph of the first created collection (if there are two collections for the same historians, only one includes data about the historian).

HANDS-ON

GET THE DATA

https://github.com/marilenadaquino/epds
Download from the folder resources/

Artchives.nq [data]
Artchives.csv [table of classes and predicates]

Create a folder dedicated to this hands-on class.

Copy artchives.nq in a subfolder called “**resources**”

You will have a structure like the following:

dhdk_epds/
resources/
artchives.nq
artchives.csv

INSTALL PACKAGES

Open the **shell/terminal**

Install rdflib python with pip with the following command:
pip install rdflib

-- if you have mac you may have to use
pip3 install rdflib

Install pprint (if you don’t have it yet)
pip install pprint

SET UP THE CODING ENVIRONMENT

Open the **editor** you chose (e.g. PyCharm)

Create a **new python file.**

Save it in a folder called “**tutorials**”, in the same main folder where the folder including the data is.

You will have a structure like the following:

dhdk_epds/
resources/
artchives.nq
artchives.csv
tutorials/
rdflib_tutorial.py

MOVE TO THE TUTORIAL

https://github.com/marilenadaquino/epds
Go to the folder tutorials/
open in the browser the file called
rdflib_tutorial.ipynb

HOMEWORK

YET ANOTHER QUESTIONNAIRE!

HERE https://forms.gle/uWW3j7sXNZ8WF3B89

SPOILER ALERT: You will need to code in order to answer the questions.

THANKS

Does anyone have any questions?

marilena.daquino2@unibo.it
github