

# Medical Informatics

## Lecture 10: More RDF

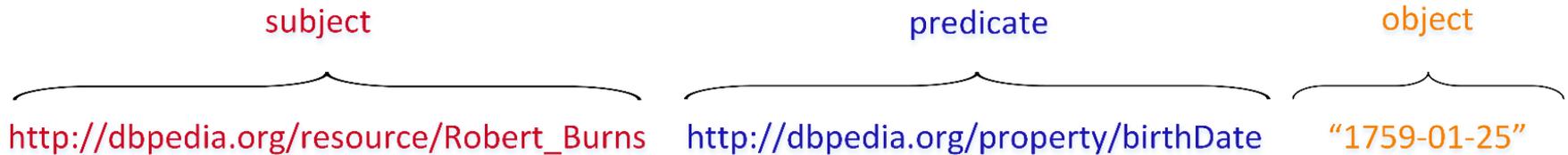
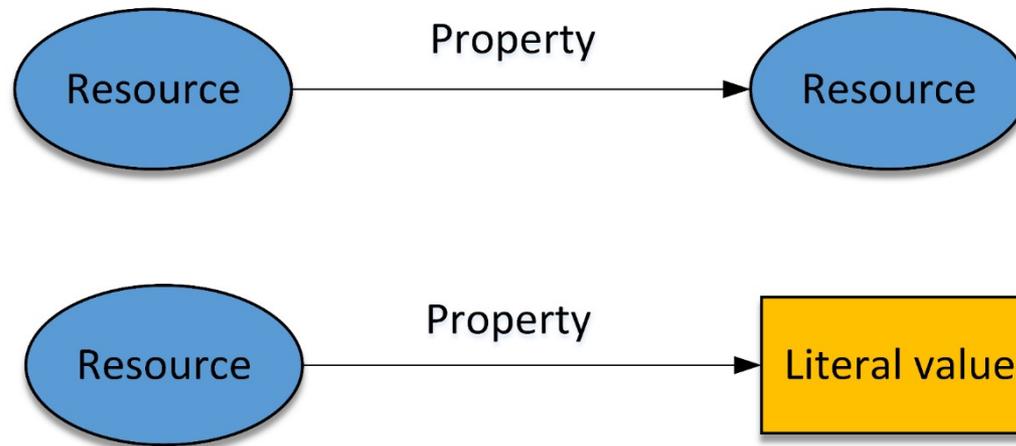
Dr Areti Manataki



Nanjing Medical University

# In the previous lecture

- RDF data model



# In the previous lecture

- In RDF we use URIs to uniquely identify resources and predicates.
- Vocabularies: FOAF (e.g. foaf:knows, foaf:name, foaf:based\_near, etc.)
- Serialisation: Turtle, XML

```
@prefix dbp: <http://dbpedia.org/property/> .
@prefix usherres: <http://usher.ed.ac.uk/medinf/resource/> .
@prefix ushervoc: <http://usher.ed.ac.uk/medinf/vocab/> .
usherres:aroast dbp:name "Artisan Roast" .
usherres:aroast dbp:locatedIn usherres:eastEdinburgh .
usherres:aroast ushervoc:stars "5" .
```

# In this lecture

- Turtle serialisation
- A short introduction to RDFS

# Turtle serialisation

# URIs and QNames in Turtle

- Full URIs are enclosed in `< >`.

```
# this is a comment
<http://dbpedia.org/resource/Robert_Burns> <http://xmlns.com/foaf/0.1/name> "Robert Burns" .
```

- QNames can be used, as long as QName-URI bindings are provided.

```
@prefix dbr: <http://dbpedia.org/resource/> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .

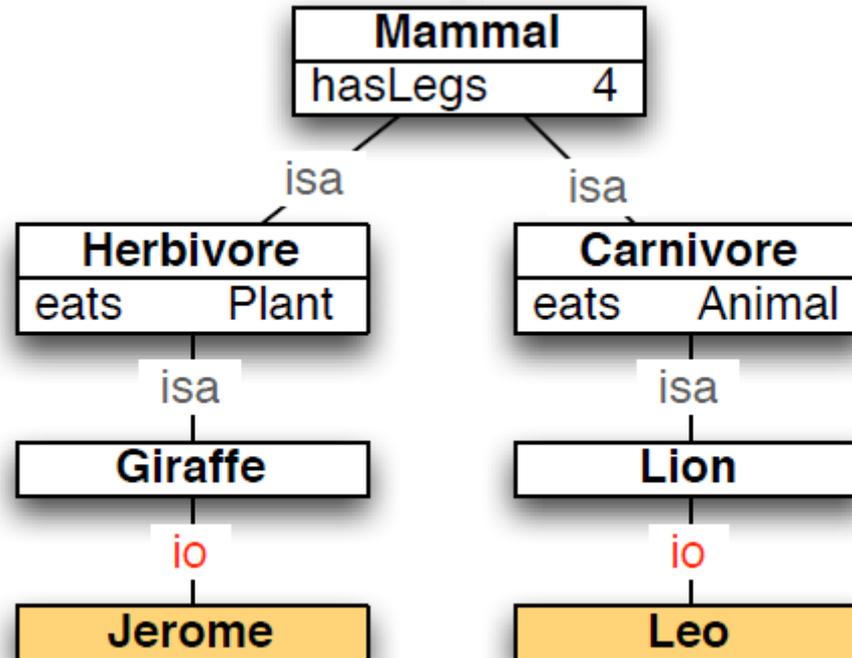
dbr:Robert_Burns foaf:name "Robert Burns" .
```

# Literals and Datatypes in Turtle

- Literals are written with double quotes.
- Typed literal values consist of a literal appended by ^^ and a URI – usually from XML Schema.
  - Example datatypes: xsd:string, xsd:integer, xsd:double, xsd:date, xsd:dateTime, xsd:boolean, xsd:decimal
- Turtle has a shorthand syntax for writing integer, decimal values, and double values.

```
@prefix dbp: <http://dbpedia.org/property/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix usherres: <http://usher.ed.ac.uk/medinf/resource/> .
@prefix ushervoc: <http://usher.ed.ac.uk/medinf/vocab/> .
usherres:aroast dbp:name "Artisan Roast"^^xsd:string .
usherres:aroast ushervoc:stars "5"^^xsd:integer .
usherres:aroast ushervoc:numberOfEmployees 12 .
```

# Instance-of in RDF



Class membership is expressed via `rdf:type`

# Instance-of in RDF

- Jerome's and Leo's class membership

```
@prefix : <http://zoo.org/> .  
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .  
:jerome rdf:type :Giraffe .  
:leo rdf:type :Lion .
```

- Leonardo DiCaprio's class membership

```
@prefix : <http://usher.ed.ac.uk/medinf/resource/> .  
@prefix ushervoc: <http://usher.ed.ac.uk/medinf/vocab/> .  
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .  
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .  
:Leonardo_DiCaprio rdf:type ushervoc:Actor .  
ushervoc:Actor rdf:type rdfs:Class .
```

# Instance-of in RDF

We use `a` as an abbreviation of `rdf:type`

```
@prefix : <http://zoo.org/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
:jerome a :Giraffe .
:leo a :Lion .

@prefix : <http://usher.ed.ac.uk/medinf/resource/> .
@prefix ushervoc: <http://usher.ed.ac.uk/medinf/vocab/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
:Leonardo_DiCaprio a ushervoc:Actor .
ushervoc:Actor a rdfs:Class .
```

# Abbreviating groups of triples

We can use `,` to abbreviate repeated subjects and predicates

- Triples about Leonardo DiCaprio, in full form.

```
dbpedia:Leonardo_DiCaprio :playedIn dbpedia:The_Wolf_of_Wall_Street .  
dbpedia:Leonardo_DiCaprio :playedIn dbpedia:Inception .
```

- Triples about Leonardo DiCaprio, in abbreviated form.

```
dbpedia:Leonardo_DiCaprio :playedIn dbpedia:The_Wolf_of_Wall_Street ,  
dbpedia:Inception .
```

# Abbreviating groups of triples

We can use `;` to abbreviate repeated subjects

- Triples about Leonardo DiCaprio, in full form.

```
dbpedia:Leonardo_DiCaprio rdf:type :Actor .  
dbpedia:Leonardo_DiCaprio :playedIn dbpedia:The_Wolf_of_Wall_Street .
```

- Triples about Leonardo DiCaprio, in abbreviated form.

```
dbpedia:Leonardo_DiCaprio rdf:type :Actor ;  
                           :playedIn dbpedia:The_Wolf_of_Wall_Street .
```

# Abbreviating groups of triples

## Combining both abbreviations

- Triples about Leonardo DiCaprio, in full form.

```
dbpedia:Leonardo_DiCaprio rdf:type :Actor .  
dbpedia:Leonardo_DiCaprio :playedIn dbpedia:The_Wolf_of_Wall_Street .  
dbpedia:Leonardo_DiCaprio :playedIn dbpedia:Inception .
```

- Triples about Leonardo DiCaprio, in abbreviated form.

```
dbpedia:Leonardo_DiCaprio rdf:type :Actor ;  
                           :playedIn dbpedia:The_Wolf_of_Wall_Street ,  
                                   dbpedia:Inception .
```

# Assertion statements

- RDF allows us to make factual statements, i.e. statements about individuals. (ABox)
- We can say that Leonardo DiCaprio is an Actor, or that he played in The Wolf of Wall Street.
- But we can't say things like:
  - Actors are artists.
  - If you are a friend of someone then you know that person.

# Introduction to RDFS

# Terminological statements with RDFS

- **RDF Schema** (RDFS) allows us to define our own vocabulary, and thus specify classes, their hierarchy and relations between them. (TBox)
- RDFS is expressed in RDF, i.e. as a set of triples.
- Basic idea is to allow statements like the following:
  - Every instance of Woman is an instance of Person.
  - The subject of 'age' must be a Agent.
  - The object of 'age' must be a literal.

# Some RDF / RDFS classes

- RDF:
  - `rdf:type` – an instance of `rdf:Property` used to state that a resource is an instance of a class
  - `rdf:Property` – the class of properties
- RDFS:
  - `rdfs:Class` – the class of classes
  - `rdfs:subClassOf` – the subject is a subclass of a class
  - `rdfs:subPropertyOf` – the subject is a subproperty of a property
  - `rdfs:domain` – a domain of the subject property
  - `rdfs:range` – a range of the subject property

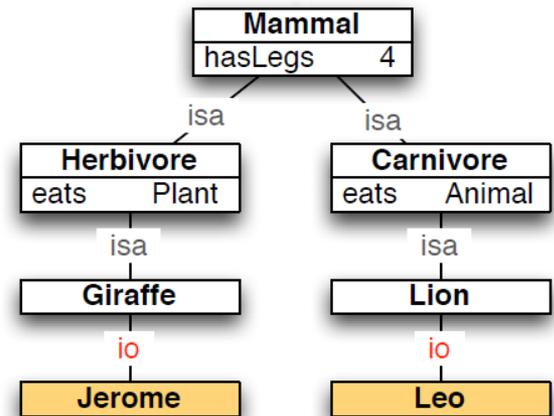
# Our animal kingdom in RDFS: classes

- Declaring classes: Giraffe and Herbivore

```
terms:Giraffe  rdf:type  rdfs:Class .
terms:Herbivore  rdf:type  rdfs:Class .
```

- Declaring instances: Jerome is an instance of Giraffe

```
myzoo:jerome  rdf:type  terms:Giraffe .
myzoo:jerome  a  terms:Giraffe .
```



- Declaring class hierarchy: Giraffe is a subclass of Herbivore

```
terms:Giraffe  rdfs:subClassOf  terms:Herbivore .
```

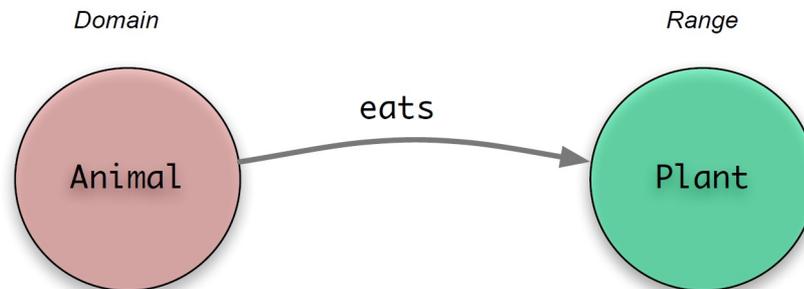
# Our animal kingdom in RDFS: properties

- Declaring properties: eats

```
terms:eats  rdf:type  rdf:Property .  
terms:eats  a  rdf:Property .
```

- Declaring property domains and ranges: animals eat plants

```
terms:eats  rdfs:domain  terms:Animal .  
terms:eats  rdfs:range  terms:Plant .
```



# RDFS provides meaning

- RDF allows us to express statements in the form of triples.
- But it has no way of telling which URIs can semantically act as predicates.
  - For example, the following is a valid RDF statement:  
`:areti terms:birthPlace "dog" .`
- RDFS helps provide meaning to RDF data.
- It allows for inference, so that you get out more than what is directly asserted.

# Type propagation in RDFS

- Jerome is a Giraffe and Giraffes are Mammals.
- Therefore Jerome is a Mammal.
- We get this with the use of the following type propagation rule:

IF

```
?A rdfs:subClassOf ?B .
```

AND

```
?x rdf:type ?A .
```

THEN

```
?x rdf:type ?B .
```

# Conclusions

- Turtle serialisation
  - Full URIs or Qnames
  - Several abbreviations
- RDFS
  - It allows us to define simple vocabularies
  - We're going to be using: `rdfs:Class`, `rdf:Property`, `rdfs:subClassOf`, `rdfs:subPropertyOf`, `rdfs:domain`, `rdfs:range`
  - In the next lecture we'll be introduced to the SPARQL query language.

# Acknowledgements

The content of these slides was originally created for the Medical Informatics course from The University of Edinburgh, which is licensed under a Creative Commons Attribution-ShareAlike 4.0 International (CC BY-SA 4.0) license.

These lecture slides are also licensed under a CC BY-SA 4.0 license.

