Week 3

RDF -2

Useful link: RDF/XML

- https://www.w3.org/TR/rdf-syntax-grammar/

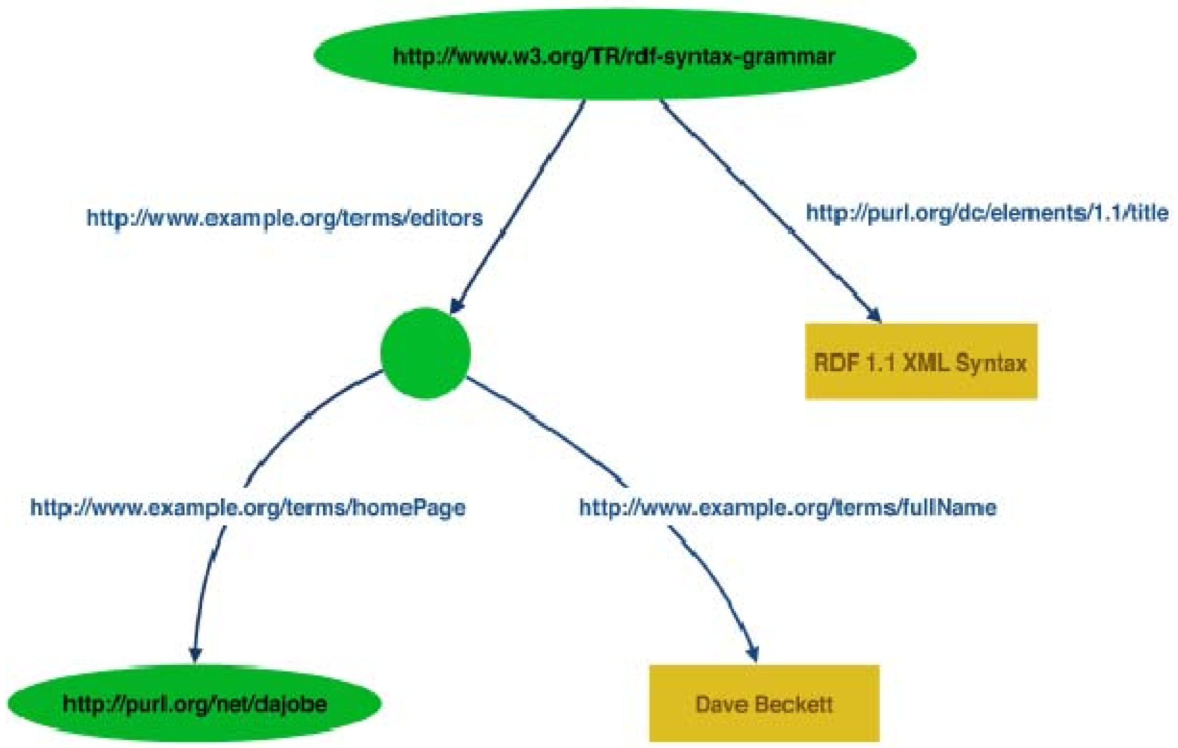
RDF/XML

- RDF document represented by XML statement with the tag `rdf:RDF`
- It is necessary to declare that RDF is being used so that applications can recognise this is an RDF/XML document.
- The content of the element is a number of descriptions which use `rdf:Description` tags
  - Every description is a statement about a resource
    - An `about` attribute, referencing an existing resource
    - An `ID` attribute, creating a new resource
    - Without a name, creating an anonymous resource

Ora Lassila is the creator of the resource http://www.w3.org/Home/Lassila

```
<?xml version="1.0"?>
<rdf:RDF
  xmlns:rdf=http://www.w3.org/1999/02/22-rdf-syntax-ns#
  xmlns:s="http://description.org/schema/">
  <rdf:Description
  about="http://www.w3.org/Home/Lassila">
      <s:Creator>Ora Lassila</s:Creator>
  </rdf:Description>
</rdf:RDF>
```

Another example: look at the green nodes


http://www.w3.org/TR/rdf-syntax-grammar
http://www.example.org/terms/editors
http://purl.org/dc/elements/1.1/title
RDF 1.1 XML Syntax
http://www.example.org/terms/homePage
http://www.example.org/terms/fullName
http://purl.org/net/dajobe
Dave Beckett

```
<rdf:Description
rdf:about="http://www.w3.org/TR/rdf-syntax-grammar">
  <ex:editor>
    <rdf:Description>
      <ex:homePage>
        <rdf:Description
rdf:about="http://purl.org/net/dajobe/">
        </rdf:Description>
      </ex:homePage>
    </rdf:Description>
  </ex:editor>
</rdf:Description>
```

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:dc="http://purl.org/dc/elements/1.1/"
 xmlns:ex="http://example.org/">
 <rdf:Description rdf:about="http://www.w3.org/TR/rdf-syntax-grammar">
 <ex:editor>
   <rdf:Description>
     <ex:homePage>
       <rdf:Description rdf:about="http://purl.org/net/dajobe/">
       </rdf:Description>
     </ex:homePage>
   </rdf:Description>
 </ex:editor>
</rdf:Description>

</rdf:RDF>
```

Blank node

- The next few slides show several abbreviations when a node element about a resource has multiple property elements

Multiple property element:
The subject node with IRI http://www.w3.org/TR/rdf-syntax-grammar has property elements ex:editor and ex:title and the node element for the blank node can take ex:homePage and ex:fullName

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:ex="http://example.org/">

<rdf:Description rdf:about="http://www.w3.org/TR/rdf-syntax-
grammar">
  <ex:editor>
    <rdf:Description>
      <ex:homePage>
        <rdf:Description rdf:about="http://purl.org/net/dajobe/">
        </rdf:Description>
      </ex:homePage>
      <ex:fullName>Dave Beckett</ex:fullName>
    </rdf:Description>
  </ex:editor>
  <dc:title>RDF 1.1 XML Syntax</dc:title>
</rdf:Description>
</rdf:RDF>
```

Empty Property element

In this example, the property element ex:homePage contains an empty node element with the IRI http://purl.org/net/dajobe/

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-
ns#"
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:ex="http://example.org/">

<rdf:Description rdf:about="http://www.w3.org/TR/rdf-syntax-
grammar">
  <ex:editor>
    <rdf:Description>
      <ex:homePage rdf:resource="http://purl.org/net/dajobe/"/>
      <ex:fullName>Dave Beckett</ex:fullName>
    </rdf:Description>
  </ex:editor>
  <dc:title>RDF 1.1 XML Syntax</dc:title>
</rdf:Description>
</rdf:RDF>
```

Property attributes: When a property element's content is string literal, it may be possible to use it as an XML attribute on the containing node element

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-
ns#"
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:ex="http://example.org/">

<rdf:Description rdf:about="http://www.w3.org/TR/rdf-syntax-
grammar"
           dc:title="RDF 1.1 XML Syntax">
  <ex:editor>
    <rdf:Description ex:fullName="Dave Beckett">
      <ex:homePage rdf:resource="http://purl.org/net/dajobe/"/>
    </rdf:Description>
  </ex:editor>
</rdf:Description>

</rdf:RDF>
```

Additional homework

- Refer to https://www.w3.org/TR/rdf-syntax-grammar/
- Read the use of
  - xml: lang (see section 2.7)
  - xml: literals (see section 2.8)
  - Typed literals (see section 2.9)
  - Identifying blank nodes (see section 2.10)
  - Omitting blank nodes (see section 2.11)
  - Omitting nodes: property attributes on an empty Property element (see section 2.12)
  - Typed node elements (see section 2.13)
  - Abbreviating URIs (see section 2.14)

Containers

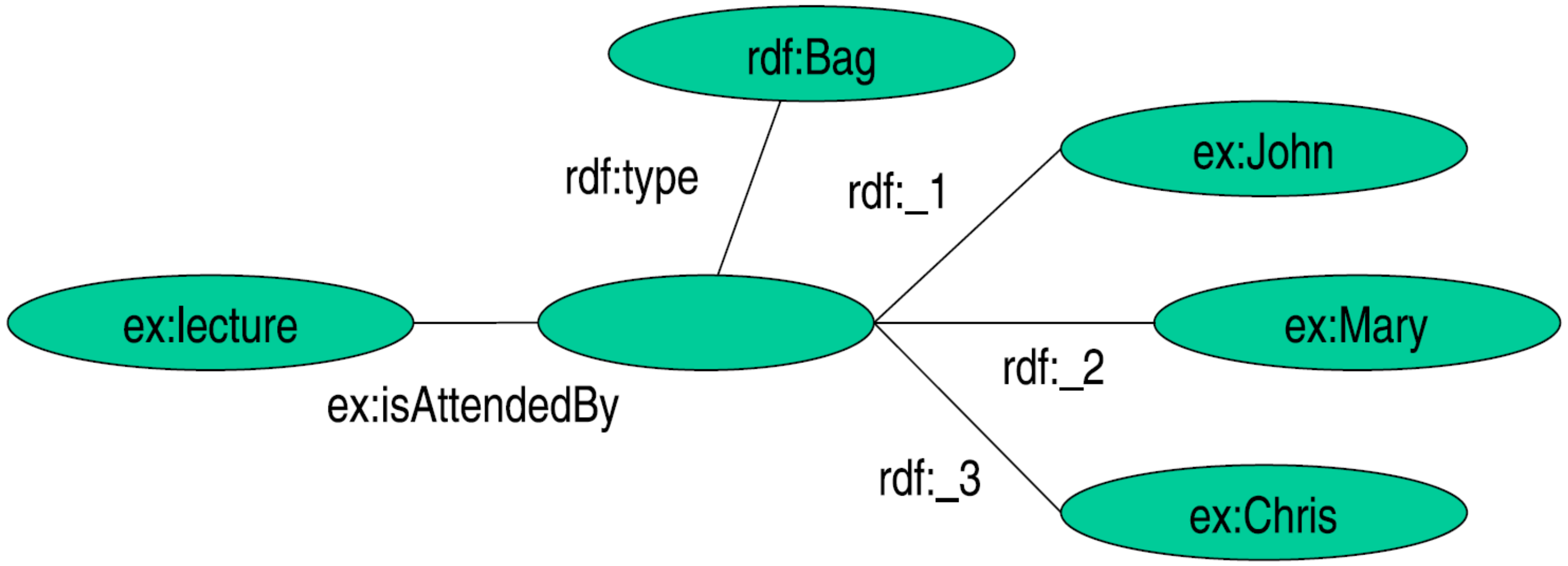
- refer to a collection of resources
  - e.g. a list of students
- three types of container objects
  - Bag (`rdf: Bag`)
  - Sequence (`rdf: Seq )`
  - Alternative (`rdf: Alt`)
- Therefore the rdfs:Container class is a super-class of rdf:Bag, rdf:Seq, rdf:Alt

`rdf:Bag`

- an unordered list of resources or literals
- to declare a property with multiple values and there is no significance to the order in which the values are given
- e.g.
  - a list of part numbers where order of processing is unimportant, duplicate values are permitted

RDF Containers Graph Representation: Bag

"The lecture is attended by John, Mary and Chris"



A list of favourite fruits: banana, apple and pear

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">

  <rdf:Bag rdf:about="http://example.org/favourite-fruit">
    <rdf:_1 rdf:resource="http://example.org/banana"/>
    <rdf:_2 rdf:resource="http://example.org/apple"/>
    <rdf:_3 rdf:resource="http://example.org/pear"/>
  </rdf:Seq>

</rdf:RDF>
```

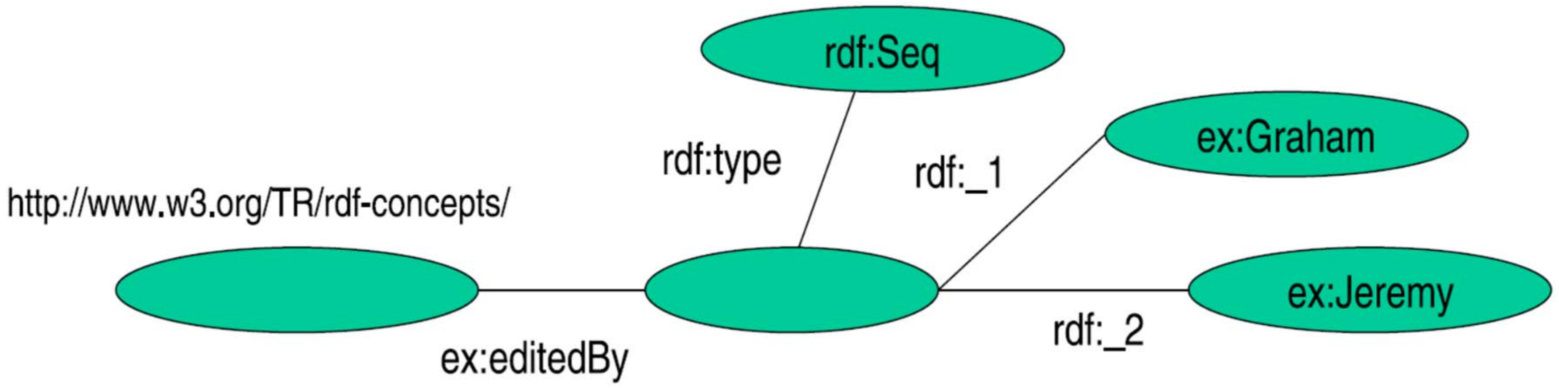
`rdf:Seq`

- an ordered list of resources or literals
- to declare a property with multiple values and order of the values is significant
- e.g.
  - alphabetical ordering of values, duplicate values are permitted

RDF Containers Graph Representation: Seq

“[RDF-Concepts] is edited by Graham and Jeremy

(<u>in that order</u>)”



A list of favourite fruits: banana, apple and pear (in the order specified)

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">

  <rdf:Seq rdf:about="http://example.org/favourite-fruit">
    <rdf:_1 rdf:resource="http://example.org/banana"/>
    <rdf:_2 rdf:resource="http://example.org/apple"/>
    <rdf:_3 rdf:resource="http://example.org/pear"/>
  </rdf:Seq>

</rdf:RDF>
```

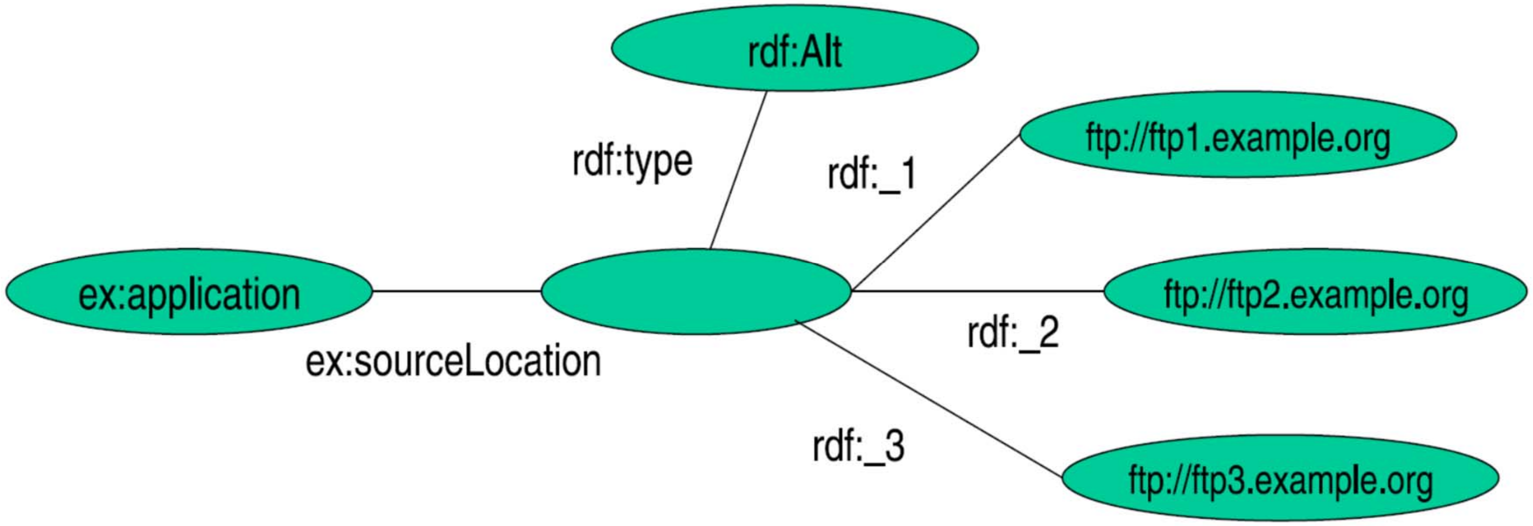
`rdf:Alt`

- a list of resources or literals for the <u>single</u> value of a property
  - e.g. provide alternative language translations for the title of the work, or to provide a list of Internet mirror sites at which the resource might be found
- can choose any one of the items in the list as appropriate

RDF Containers Graph Representation: Alt

"The source code for the application may be found at

ftp1.example.org, ftp2.example.org, ftp3.example.org"



A list of favourite fruits: banana, apple and pear (choose one from the list)

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-
  rdf-syntax-ns#">

  <rdf:Alt rdf:about="http://example.org/favourite-
  fruit">
    <rdf:_1
  rdf:resource="http://example.org/banana"/>
    <rdf:_2
  rdf:resource="http://example.org/apple"/>
    <rdf:_3 rdf:resource="http://example.org/pear"/>
  </rdf:Seq>

</rdf:RDF>
```

`rdf:li`

- a convenient element to avoid having to explicitly number each member
  - list item

A list of favourite fruits: banana, apple and pear

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-
   ns#">

  <rdf:Seq rdf:about="http://example.org/favourite-fruit">
    <rdf:li rdf:resource="http://example.org/banana"/>
    <rdf:li rdf:resource="http://example.org/apple"/>
    <rdf:li rdf:resource="http://example.org/pear"/>
  </rdf:Seq>

</rdf:RDF>
```

Predicate Lists in N-Triple

- Often the same subject will be referenced by a number of predicates.
- use the '**;**' symbol to repeat the subject of triples that vary only in predicate and object RDF terms

Example

```
<http://example.org/#spiderman>
<http://www.perceive.net/schemas/relationship/enemyOf>
<http://example.org/#green-goblin> ;
<http://xmlns.com/foaf/0.1/name> "Spiderman" .
```

Object list in N-Triple

- Objects are repeated with the same subject and predicate.
- the ',' symbol is used to repeat the subject and predicate of triples that only differ in the object RDF term.

Example

```
<http://example.org/#spiderman>
<http://xmlns.com/foaf/0.1/name>
"Spiderman", "Человек-паук"@ru .
```

Turtle (Terse RDF Triple Language)

- a more compact serialization of RDF
- uses prefix
- A *prefixed name* is a prefix label and a local part, separated by a colon ":"

Example

```
@base <http://example.org/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix rel: <http://www.perceive.net/schemas/relationship/> .

<#green-goblin>
   rel:enemyOf <#spiderman> ;
   a foaf:Person ;    # in the context of the Marvel universe
   foaf:name "Green Goblin" .

<#spiderman>
   rel:enemyOf <#green-goblin> ;
   a foaf:Person ;
   foaf:name "Spiderman", "Человек-паук"@ru .
```

Example

- Define a prefix label

  `http://www.perceive.net/schemas/relationship/` as `somePrefix`

Then write

```
somePrefix:enemyOf
```

is equivalent to

```
<http://www.perceive.net/schemas/relationship/enemyOf>
```

Homework

- Check Example 9 at https://www.w3.org/TR/turtle/

to look at different ways of writing IRIs in Turtle

RDF Literals

```
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
<http://example.org/#green-goblin> foaf:name
"Green Goblin" .
<http://example.org/#spiderman> foaf:name
"Spiderman" .
```

Homework

- Check Example 11, 12, 13 at https://www.w3.org/TR/turtle/

to look at Quoted Literals, Numbers, Booleans respectively

RDF Blank Nodes

- In Turtle
  - expressed as _: followed by a blank node label which is a series of name characters.
- A fresh RDF blank node is allocated for each unique blank node label in a document. Repeated use of the same blank node label identifies the same RDF blank node.

Example

```
_:a <http://xmlns.com/foaf/0.1/name> "Alice" .
_:a <http://xmlns.com/foaf/0.1/knows> _:b .
_:b <http://xmlns.com/foaf/0.1/name> "Bob" .
_:b <http://xmlns.com/foaf/0.1/knows> _:c .
_:c <http://xmlns.com/foaf/0.1/name> "Eve" .
_:b <http://xmlns.com/foaf/0.1/mbox> <bob@example.com> .
```

Collections

- Collection structure for lists of RDF nodes
- The Turtle syntax for Collections is a possibly empty list of RDF terms enclosed by ()
- Reference:
  - https://www.w3.org/TR/rdf-schema/#ch_containervocab

Example

```
@prefix : <http://example.org/foo> .
# the object of this triple is the RDF collection
blank node
:subject :predicate ( :a :b :c ) .

# an empty collection value - rdf:nil
:subject :predicate2 () .
```

RDF Collection

- rdf:List
- rdf:first
- rdf:rest
- rdf:nil
- Reference
  https://www.w3.org/TR/turtle/#collections

The RDF Schema (RDFS)

- Link: https://www.w3.org/TR/rdf-schema/
- Is a semantic extension of RDF
  - May impose special syntactic conditions or restrictions upon RDF graphs
- It provides mechanisms for describing groups of related resources and the relationships between these resources
  - e.g. we could define the `eg:author` property to have a **domain** of `eg:Document` and a **range** of `eg:Person`

Example1

- Types in RDF:

  <#john, rdf:type, #Student>

- What is a “**#Student**”?
  - “**#Student**” identifies a category (a concept or a class)

- We need a language for defining RDF types:
  - Define classes:
    - *“**#Student** is a class”*
  - Relationships between classes:
    - *“**#Student** is a sub-class of **#Person**”*
  - Properties of classes:
    - *“**#Person** has a property **hasName**”*

- RDF Schema is such a language

RDFS: Class & Property

- RDF Schema describes properties in terms of the classes of resource to which they apply.
- This is the role of the domain and range mechanisms
- Example,
  - `eg:author` property has a **domain** of `eg:Document` and a **range** of `eg:Person`,
  - whereas a classical object oriented system may define a class eg:Book with an attribute called eg:author of type eg:Person.
  - Using the RDF approach, it is easy for others to subsequently define additional properties with a domain of eg:Document or a range of eg:Person. This can be done without the need to re-define the original description of these classes.
  - One benefit of the RDF property-centric approach is that it allows anyone to extend the description of existing resources, one of the architectural principles of the Web
- RDFS strategy is to acknowledge that there are many techniques through which the meaning of classes and properties can be described

RDFS Vocabulary

- RDFS Extends the RDF Vocabulary
- RDFS summary can be found at the following link and https://www.w3.org/TR/rdf-schema/#ch_summary
- Namespace
  rdfs: https://www.w3.org/TR/rdf-schema#

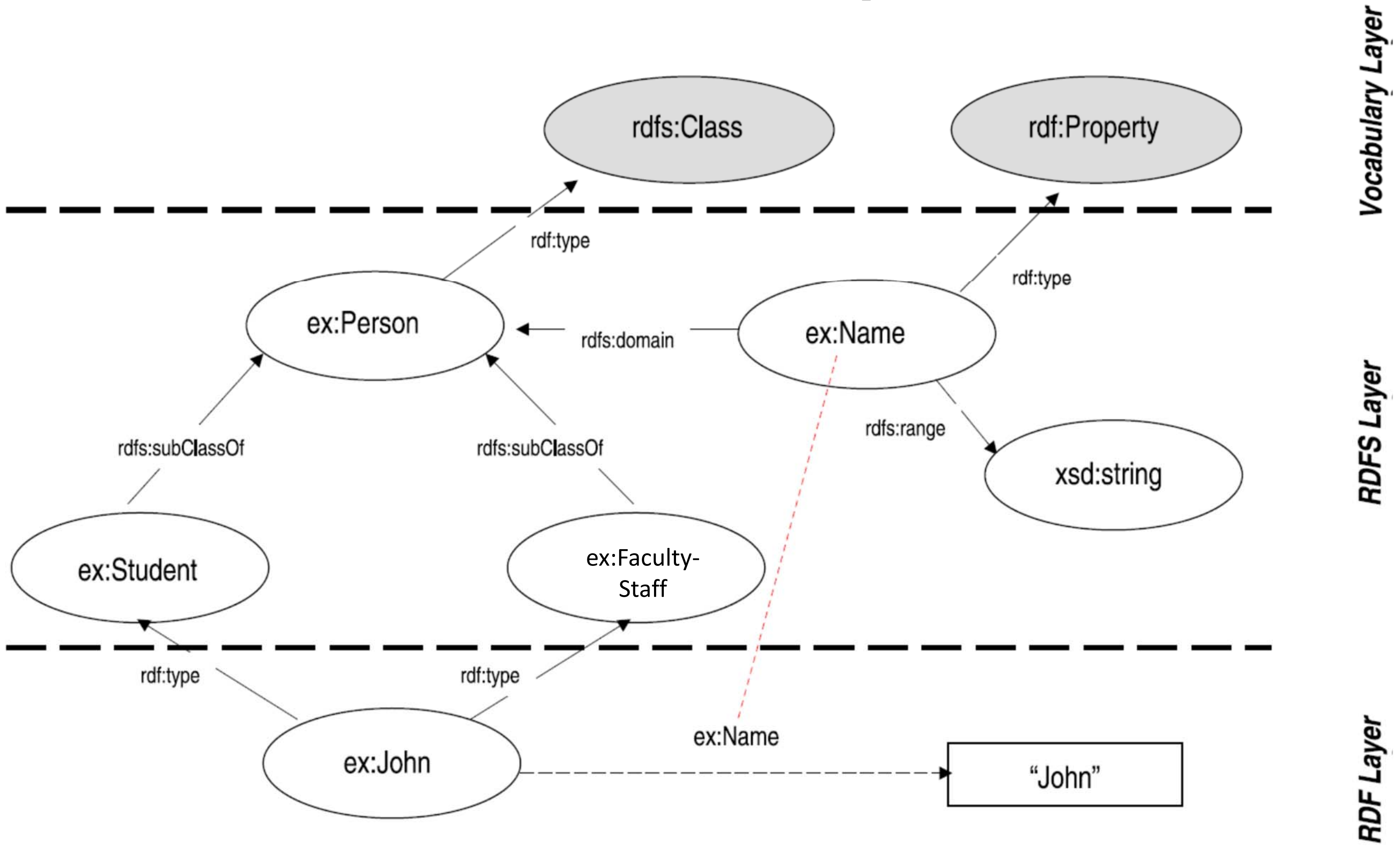
RDFS Classes

- `rdfs:Resource`
- `rdfs:Class`
- `rdfs:Literal`
- `rdfs:Datatype`
- `rdfs:Container`
- `rdfs:ContainerMembershipProperty`

RDFS Properties

- `rdfs:domain`
- `rdfs:range`
- `rdfs:subPropertyOf`
- `rdfs:subClassOf`
- `rdfs:member`
- `rdfs:seeAlso`
- `rdfs:isDefinedBy`
- `rdfs:comment`
- `rdfs:label`

RDFS Example



Classes

- Resources may be divided into groups called classes.
- The members of a class are known as *instances* of the class.

```
rdfs: Class
```

Subclass

- If a class C is a *subclass* of a class C', then all instances of C will also be instances of C'.

`rdfs:subClassOf`

Property

- property → characteristics of class
- rdf: Property
  - all properties in RDF are instances of class rdf:Property
  - example: ex:age rdf:type rdf:Property
- To describe property
  - rdfs: domain
  - rdfs:range
  - rdfs:subPropertyOf

rdfs:range

- the values of a particular property
- example

```
ex:hasMother rdfs:range ex:Female .
ex:age rdfs:range xsd:integer.
```

rdfs:domain

- a particular property applies to a designated class.

```
ex:Book rdf:type rdfs:Class .
ex:author rdf:type rdf:Property .
ex:author rdfs:domain ex:Book .
```

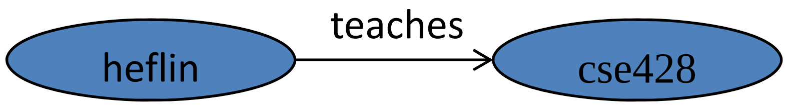
Example

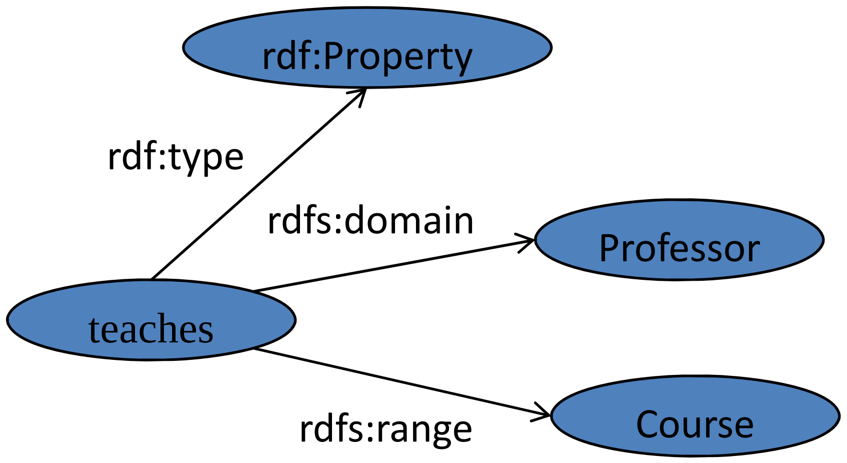
```
<rdf:Property rdf:ID="registeredTo">
  <rdfs:domain rdf:resource="#MotorVehicle"/>
  <rdfs:range rdf:resource="#Person"/>
</rdf:Property>

<rdf:Property rdf:ID="rearSeatLegRoom">
  <rdfs:domain rdf:resource="#PassengerVehicle"/>
  <rdfs:range rdf:resource="&xsd;integer"/>
</rdf:Property>
```

RDF Schema Example

```
<rdf:RDF xml:base="http://example.org/univ-ont#"
       xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
       xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
       xmlns:univ="http://example.org/univ-ont#">
   <rdf:Property rdf:about="#teaches">
       <rdfs:domain rdf:resource="#Professor" />
       <rdfs:range rdf:resource="#Course" />
   </rdf:Property>
   <univ:Person rdf:about="#heflin" >
       <univ:teaches rdf:resource="#cse428" />
   </univ:Person>
</rdf:RDF>
```





rdfs:subPropertyOf

```
ex:driver rdf:type rdf:Property .
ex:primaryDriver rdf:type rdf:Property .
ex:primaryDriver rdfs:subPropertyOf ex:driver .
```

RDF/XML

```
<rdf:Property rdf:ID="driver">
  <rdfs:domain rdf:resource="#MotorVehicle"/>
</rdf:Property>

<rdf:Property rdf:ID="primaryDriver">
  <rdfs:subPropertyOf rdf:resource="#driver"/>
</rdf:Property>
```

Example of Instance

```
<ex:PassengerVehicle rdf:ID="johnSmithsCar">
   <ex:registeredTo
   rdf:resource="http://www.example.org/staffid/85740"/>
<ex:rearSeatLegRoom
   rdf:datatype="&xsd;integer">127</ex:rearSeatLegRoom>
<ex:primaryDriver
   rdf:resource="http://www.example.org/staffid/85740"/>
   </ex:PassengerVehicle>
```