



University of  
**Southampton**

# Resource Description Framework

COMP6256 Knowledge Graphs for AI Systems

Dr Nicholas Gibbins – [nmg@ecs.soton.ac.uk](mailto:nmg@ecs.soton.ac.uk)

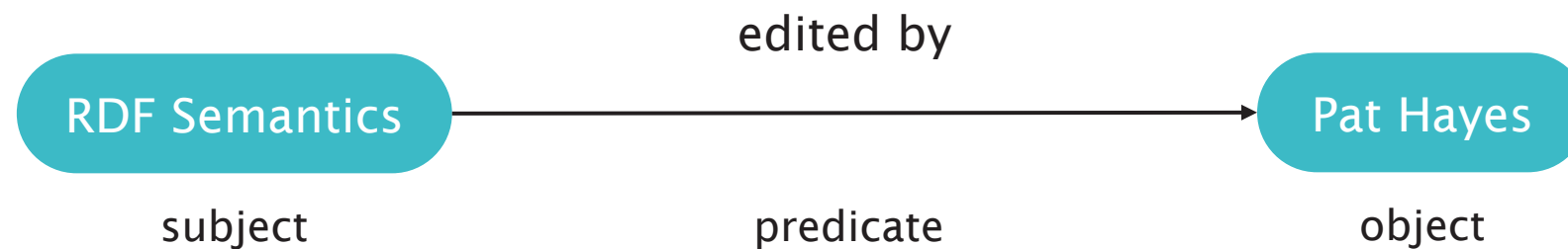
# What is the Resource Description Framework?

- A standard data model for the Semantic Web
- A knowledge representation language
- A family of data formats and notations

# A data model

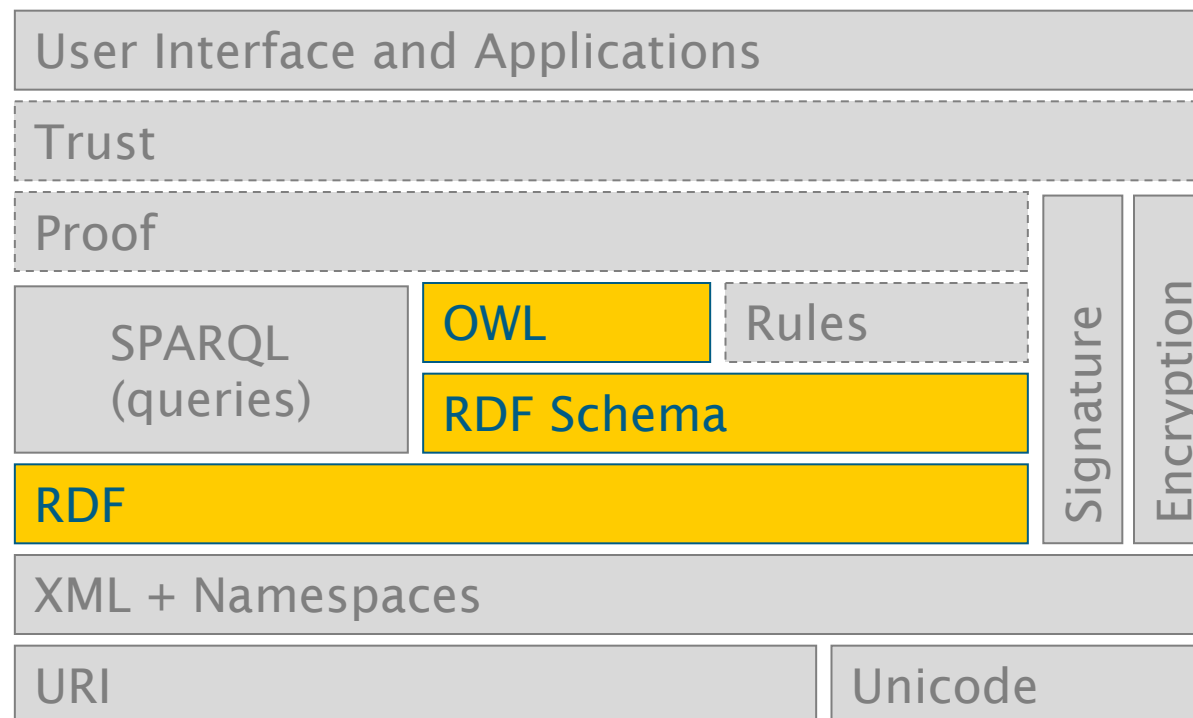
Statements in the form of subject-predicate-object **triples**

- Typed relationships between resources
- Collection of RDF statements represents a labeled, directed graph



# A knowledge representation language

RDF is used as the foundation for the other knowledge representation and ontology languages on the Semantic Web



# A family of data formats

RDF/XML is the oldest syntax

- Supported by almost all tools
- Not in any way human-friendly

RDF/N3 family

- Compact, human-friendly, non-XML syntaxes: N3, NTriples, Turtle
- We'll concentrate on Turtle in this module

Other XML and non-XML syntaxes exist:

- TriX, JSON-LD, etc

# RDF Requirements

# RDF requirements

- A means for identifying objects and vocabulary terms
- A means for distinguishing between terms from different vocabularies
- A means for serialising triples



# URIs and URIsrefs

## Standard identifiers for the Semantic Web

- Uniform Resource Identifiers are defined by RFC3986
  - `http://example.org/`
  - `urn:isbn:0198537379`
  - `mailto:nmg@ecs.soton.ac.uk`
- URI references (URIsrefs) are URIs with optional fragment identifiers
  - `http://example.org/index.html#Introduction`
  - `http://www.w3.org/1999/02/22-rdf-syntax-ns#type`

# Namespaces and qualified names

- RDF syntaxes use namespaces to abbreviate URIs to qualified names (QNames)
- A QName consists of a namespace prefix, a colon and a local name
  - e.g. `rdf:type`, `dc:creator`, `foaf:Person`
- Namespace prefixes correspond to URI prefixes

For example:

- Given a namespace prefix of `rdf` for the URI `http://www.w3.org/1999/02/22-rdf-syntax-ns#`
- The QName `rdf:type` would expand to `http://www.w3.org/1999/02/22-rdf-syntax-ns#type`

# Turtle – Terse RDF Triple Language

# The N3 family of RDF syntaxes

RDF/N3 notation designed by TimBL

- Concise format suitable for writing by hand

RDF/Ntriples defined as part of SPARQL standardisation

- Used to write test cases for compliance testing

RDF/Turtle defined as a simplified version of N3

- More faithful to RDF data model

- Resource URIs are written in angle brackets: `<http://example.org>`
- Literal values are written in double quotes: `"like this"`
- Triples terminated with a full stop: `.`
- Whitespace not relevant

# Triples in Turtle

Literals as objects:

```
<http://www.sciam.com> <http://purl.org/dc/elements/1.1/title> "Scientific American" .
```

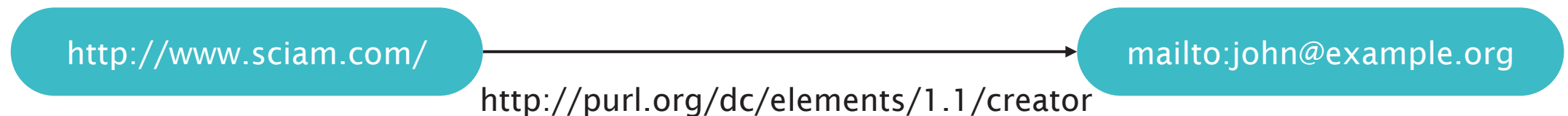
subject predicate object



# Triples in Turtle

Resources as objects:

```
<http://www.sciam.com> <http://purl.org/dc/elements/1.1/creator> <mailto:john@example.org> .
```




# Namespaces in Turtle

Defined using @prefix

```
@prefix dc: <http://purl.org/dc/elements/1.1/> .
```

```
<http://www.sciam.com> dc:creator <mailto:john@example.org> .
```

  
note: no angle brackets

http://www.sciam.com/

http://purl.org/dc/elements/1.1/creator

mailto:john@example.org

# Base URI

@base introduces a base URI relative to which all URI fragment identifiers are expanded (similar to use of xml:base in RDF/XML):

```
@base <http://example.org/data> .  
@prefix foaf: <http://xmlns.com/foaf/0.1/> .  
<#john> foaf:name "John Smith" .
```

contains the triple:

```
<http://example.org/data#john> <http://xmlns.com/foaf/0.1/name> "John Smith" .
```

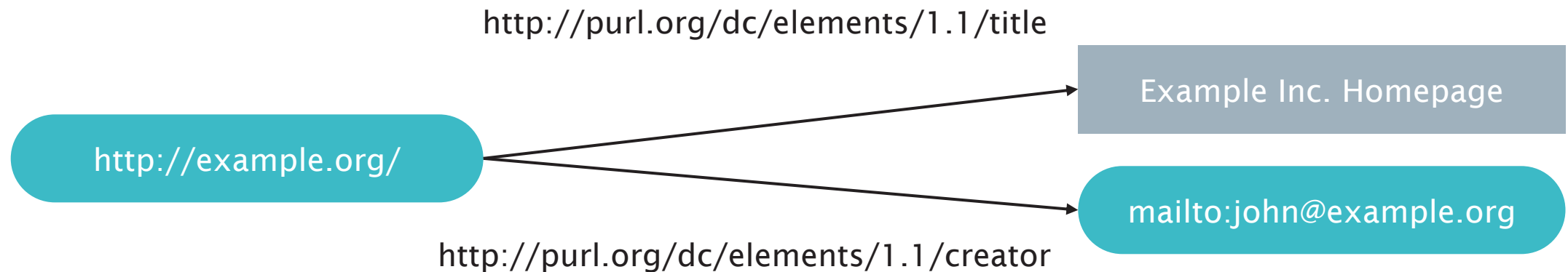


# Repeated subjects

Use semicolon ;

```
@prefix dc: < http://purl.org/dc/elements/1.1/> .
```

```
<http://example.org> dc:title "Example Inc. Homepage" ;  
                      dc:creator <mailto:john@example.org> .
```

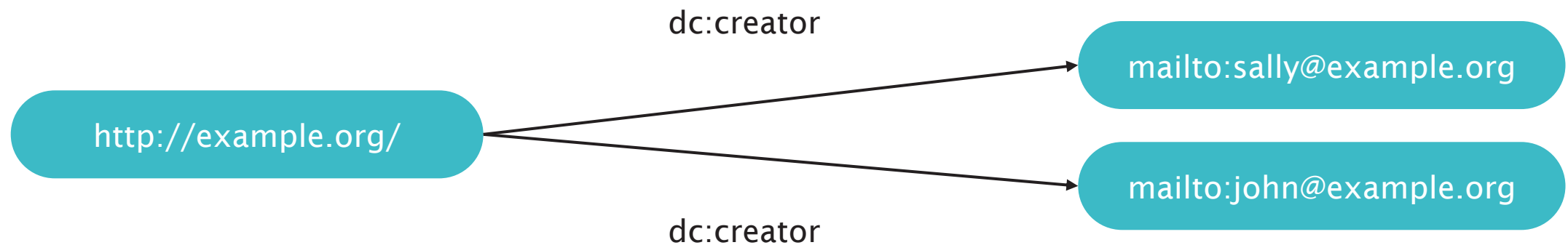


# Repeated subjects and predicates

Use comma ,

```
@prefix dc: < http://purl.org/dc/elements/1.1/> .
```

```
<http://example.org> dc:creator <mailto:john@example.org> ,  
                        <mailto:sally@example.org> .
```



# Blank nodes (bNodes)

Sometimes we have resources which we do not wish to identify with a URI

- These are *blank nodes* or *anonymous resources*

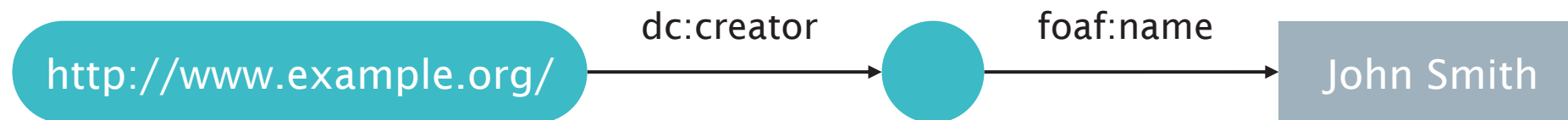


# Blank nodes (bNodes)

```
@prefix dc: < http://purl.org/dc/elements/1.1/> .
```

```
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
```

```
<http://www.example.org/> dc:creator [ foaf:name "John Smith" ] .
```

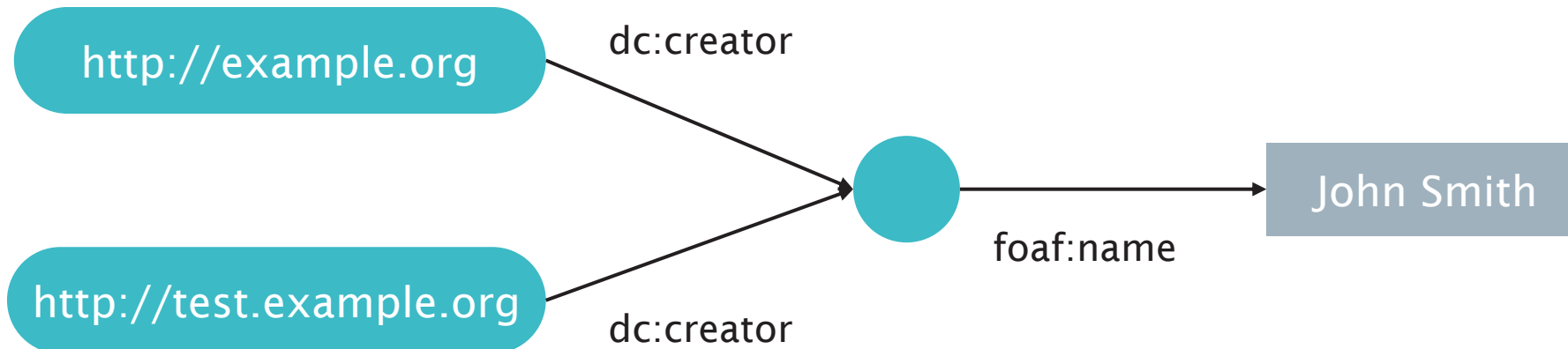


# Blank nodes (bNodes)

The [] syntax is insufficient to represent all graphs containing blank nodes unambiguously:

```
@prefix dc: < http://purl.org/dc/elements/1.1/> .  
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
```

```
<http://example.org/> dc:creator [ foaf:name "John Smith" ] .  
<http://test.example.org/> dc:creator [ foaf:name "John Smith" ] .
```



# Blank nodes and node IDs

Ambiguities resulting from blank nodes can be resolved by using node IDs

Node IDs are identifiers which are local to a given serialisation of an RDF graph

- Node IDs look like qNames with a namespace prefix of `_` (underscore)
- e.g. `_:a123` `_:foo` `_:bar`
- Node IDs may not be referred to from outside the scope of the defining graph

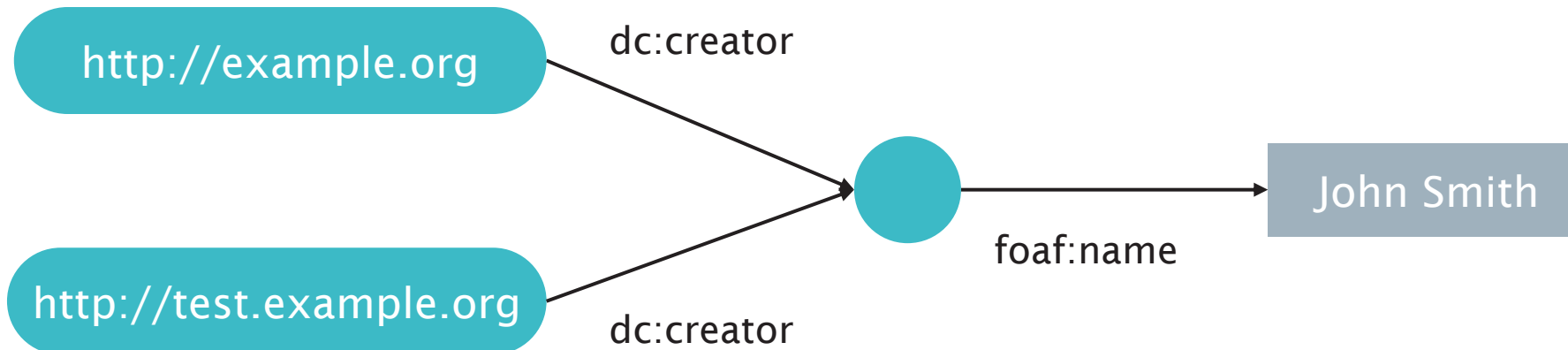
Node IDs are not guaranteed to remain unchanged when an RDF file is parsed and serialised

- The identifier strings may change  
but
- The graph structure will remain unchanged

# Blank nodes (bNodes)

```
@prefix dc: < http://purl.org/dc/elements/1.1/> .  
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
```

```
<http://example.org/> dc:creator _:foo .  
<http://test.example.org/> dc:creator _:foo .  
_:foo foaf:name "John Smith" .
```



# Datatypes

Literal values presented so far are plain and do not have a type

- Many applications need to be able to distinguish between different typed literals
- e.g. integer vs. date vs. decimal

RDF uses XML Schema datatypes:

```
@prefix dc: < http://purl.org/dc/elements/1.1/> .
```

```
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
```

```
<http://example.org/> dc:date "2020-01-27"^^xsd:date .
```



# Multilingual support

RDF supports language annotations on literals

- Languages identified by ISO369-1 two letter codes: en, zh, fr, de, es

```
@prefix dc: < http://purl.org/dc/elements/1.1/> .
```

```
<http://example.org/foreword> dc:title "Foreword"@en .
```

```
<http://example.org/foreword> dc:title "Avant-propos"@fr .
```

# Class membership

An object's membership of a class is indicated using the `rdf:type` property

Turtle lets us abbreviate `rdf:type` with 'a':

```
@prefix ex: <http://example.org/ontology/> .
```

```
<http://example.org/> a ex:website
```



# The null URI

Assertions about the null URIref `<>` are about the RDF graph itself

```
@prefix dc: < http://purl.org/dc/elements/1.1/> .
```

```
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
```

```
<> dc:date "2022-02-03"^^xsd:date ;  
    dc:creator <mailto:nmg@ecs.soton.ac.uk> .
```

# Collections

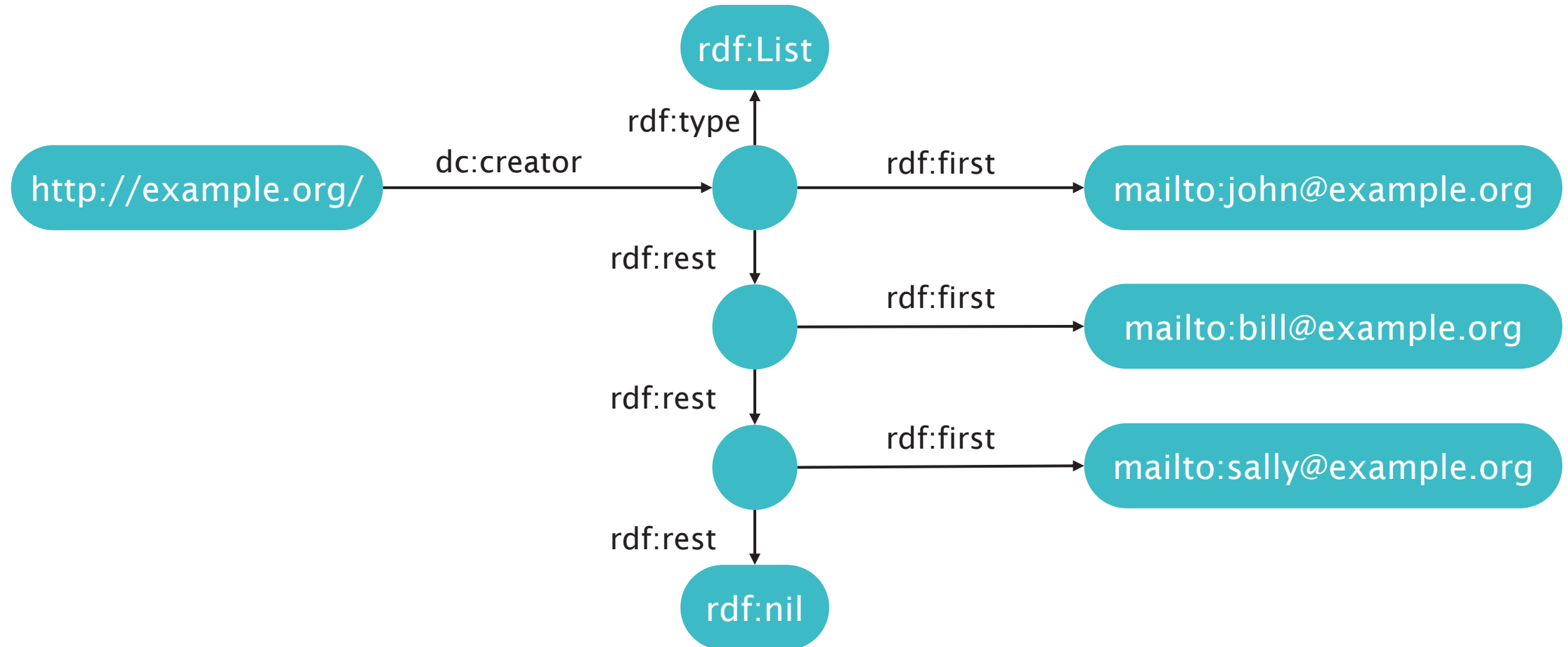
Collections (`rdf:List`) are a way of expressing ordered groups in RDF

- Recursive definition – a collection consists of the first item in the collection and another collection which is the remainder
- Resource to represent an empty collection – `rdf:nil`
- Similar to `cons/car/cdr` lists in Lisp
  - `rdf:first` equivalent to `car` – the first item in a collection
  - `rdf:rest` equivalent to `cdr` – the rest of the collection

Immutable – cannot be altered without rendering the collection ill-formed

- Relatively uncommon in plain RDF, but used extensively in the RDF serialisation of OWL (see later lecture)

# Collections



# Collections

Turtle syntax for collections uses parentheses ( ):

```
@prefix dc: <http://purl.org/dc/elements/1.1/> .
```

```
<http://example.org> dc:creator ( <mailto:john@example.org>  
                                <mailto:bill@example.org>  
                                <mailto:sally@example.org> ) .
```

# Collections

Possible to write collections without using this syntax (but why would you?)

```
@prefix dc: <http://purl.org/dc/elements/1.1/> .
```

```
<http://example.org> dc:creator [ rdf:type rdf:List ;  
                                rdf:first <mailto:john@example.org> ;  
                                rdf:rest  
                                [ rdf:first <mailto:bill@example.org> ;  
                                rdf:rest  
                                [ rdf:first <mailto:sally@example.org> ;  
                                rdf:rest rdf:nil ] ] ] .
```

# Collections

Note that these two graphs say fundamentally different things.

In this, `http://example.org/` has one creator, which is a collection:

```
@prefix dc: <http://purl.org/dc/elements/1.1/> .
```

```
<http://example.org> dc:creator ( <mailto:john@example.org>  
                                <mailto:bill@example.org>  
                                <mailto:sally@example.org> ) .
```

In this, `http://example.org/` has three creators:

```
@prefix dc: <http://purl.org/dc/elements/1.1/> .
```

```
<http://example.org> dc:creator <mailto:john@example.org> ,  
                                <mailto:bill@example.org> ,  
                                <mailto:sally@example.org> .
```



# Further Reading

# RDF Status

- Original version published in 1999
- Working group (RDF Core) formed in April 2001
- Revised version published in early 2004
- New RDF working group from in 2011 until 2013
  - New standard syntaxes (Turtle, JSON)
  - Multiple graphs and graph stores

# RDF references

- RDF homepage at W3C
  - <http://www.w3.org/RDF/>
- RDF 1.1 Primer
  - <https://www.w3.org/TR/rdf11-primer/>
- Turtle specification
  - <https://www.w3.org/TR/turtle/>
- RDF/N3 Primer
  - <http://www.w3.org/2000/10/swap/Primer.html>
- XML Schema Part 2: Datatypes
  - <http://www.w3.org/TR/xmlschema-2/>

Next Lecture: Linked Data