TECHNISCHE UNIVERSITÄT DRESDEN

FOUNDATIONS OF SEMANTIC WEB TECHNOLOGIES

Introduction to RDF

Sebastian Rudolph

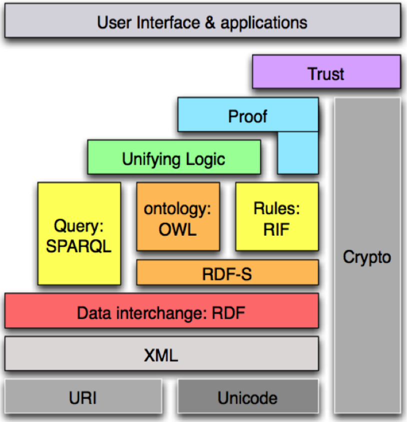
Dresden, 11 Apr 2014

DRESDEN concept
Exzellenz aus Wissenschaft und Kultur

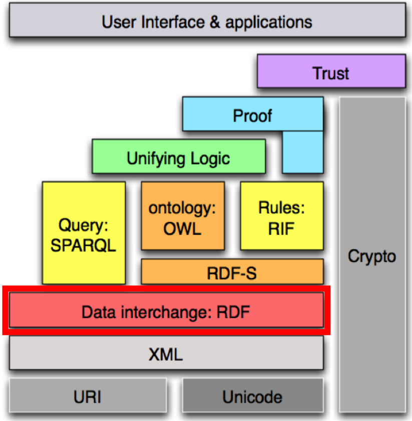
TECHNISCHE UNIVERSITÄT DRESDEN

Content

Introduction to RDF



Introduction to RDF



Agenda

- XML – Motivation
- RDF data model
- Syntax for RDF: Turtle and XML
- Datatypes
- Multi-Valued Relationships
- Blank Nodes
- Lists
- Graph Definitions
- RDF in Practice

Agenda

- XML – Motivation
- RDF data model
- Syntax for RDF: Turtle and XML
- Datatypes
- Multi-Valued Relationships
- Blank Nodes
- Lists
- Graph Definitions
- RDF in Practice

Disadvantages of XML

- tag names ambiguous (can be tackled by name spaces and URIs)
- tree structure not optimal for
  - intuitive description of the data
  - information integration
- Example: how to encode in a tree the sentence:
  "The book 'Semantic Web – Grundlagen' was published by Springer-Verlag."

Modeling Problems in XML

"The book 'Semantic Web – Grundlagen' was published by Springer-Verlag."

Modeling Problems in XML

"The book 'Semantic Web – Grundlagen' was published by Springer-Verlag."

```
<Published>
 <Publisher>Springer-Verlag</Publisher>
 <Book>Semantic Web -- Grundlagen</Book>
</Published>
```

Modeling Problems in XML

"The book 'Semantic Web – Grundlagen' was published by Springer-Verlag."

```
<Published>
 <Publisher>Springer-Verlag</Publisher>
 <Book>Semantic Web -- Grundlagen</Book>
</Published>
```

```
<Publisher Name="Springer-Verlag">
 <Published Book="Semantic Web -- Grundlagen"/>
</Publisher>
```

Modeling Problems in XML

"The book 'Semantic Web – Grundlagen' was published by Springer-Verlag."

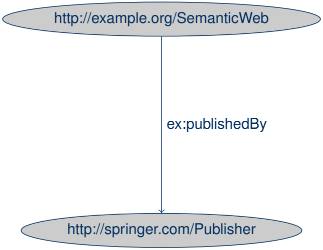
```
<Published>
 <Publisher>Springer-Verlag</Publisher>
 <Book>Semantic Web -- Grundlagen</Book>
</Published>

<Publisher Name="Springer-Verlag">
 <Published Book="Semantic Web -- Grundlagen"/>
</Publisher>

<Book Name="Semantic Web -- Grundlagen">
 <Publisher Publisher="Springer-Verlag"/>
</Book>
```

RDF: Graphs instead of Trees

Solution: Representation as (directed) Graphs



Agenda

- XML – Motivation
- RDF data model
- Syntax for RDF: Turtle and XML
- Datatypes
- Multi-Valued Relationships
- Blank Nodes
- Lists
- Graph Definitions
- RDF in Practice

General Remarks about RDF

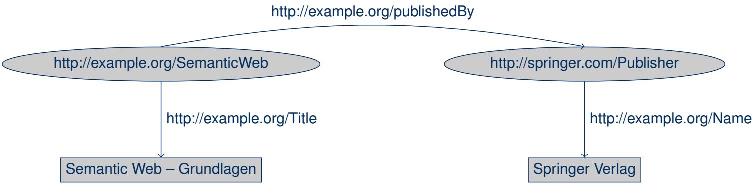
- “Resource Description Framework”
- W3C Recommendation (http://www.w3.org/RDF)
- currently being revised
- RDF is a data model
  - originally: assign metadata to Web resources, later more general usage
  - encodes structured information
  - universal, machine-readable exchange format

Constituents of RDF Graphs

- URIs
  - for uniquely referencing resources
  - (already discussed at XML lecture)
- literals
  - describe data values that do not have an independent existence
- blank nodes
  - allow for stating the existence of some individual (and describing its properties) without giving it a name

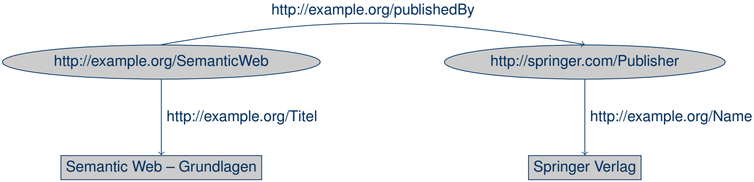
Literals

- for representing data values
- noted as strings
- interpreted by an associated datatype
- literals without datatype are treated like strings



Graph as a Set of Triples

- there are several different ways to represent graphs
- we use list of (node-edge-node) triples



RDF Triple

Constituents of an RDF triple

- inspired by linguistic but not always an exact match
- permitted occurrences of constituents:
  subject : URI or blank node
  predicate: URI (also called properties)
  object: URI or blank node or literal
- node and edge labels are unique, thus the original graph can be reconstructed from the list of triples

Agenda

- XML – Motivation
- RDF data model
- Syntax for RDF: Turtle and XML
- Datatypes
- Multi-Valued Relationships
- Blank Nodes
- Lists
- Graph Definitions
- RDF in Practice

Simple Syntax for RDF

- direct enumeration of triples:
  - N3: “Notation 3” – comprehensive formalism
  - N-Triples: fraction of N3
  - Turtle: extension of N-Triples (by abbreviations)
- Turtle syntax:
  - URIs in angular brackets
  - literals in quotes
  - tripels terminated by full stop
  - spaces and line breaks outside such delimiters are ignored

Turtle Syntax: Abbreviations

Example

```
<http://ex.org/SemanticWeb> <http://ex.org/publishedBy>
  <http://springer.com/Publisher> .
<http://ex.org/SemanticWeb> <http://ex.org/Titel>
  "Semantic Web -- Grundlagen" .
<http://springer.com/Verlag> <http://ex.org/Name>
  "Springer Verlag" .
```

In Turtle we can define prefix abbreviations:

```
@prefix ex:  <http://ex.org/> .
@prefix springer:  <http://springer.com/> .
ex:SemanticWeb ex:publishedBy springer:Publisher .
ex:SemanticWeb ex:Title "Semantic Web -- Grundlagen" .
springer:Publisher ex:Name "Springer Verlag" .
```

TECHNISCHE UNIVERSITÄT DRESDEN

Turtle Syntax: Abbreviations

Multiple triples with the same subject can be grouped:

```
@prefix ex:  <http://ex.org/> .
@prefix springer:  <http://springer.com/> .

ex:SemanticWeb     ex:publishedBy springer:Publisher ;
                   ex:Title       "Semantic Web -- Grundlagen" .
springer:Publisher ex:Name        "Springer Verlag" .
```

TECHNISCHE UNIVERSITÄT DRESDEN

Turtle Syntax: Abbreviations

Multiple triples with the same subject can be grouped:

```
@prefix ex:  <http://ex.org/> .
@prefix springer:  <http://springer.com/> .

ex:SemanticWeb     ex:publishedBy springer:Publisher ;
                   ex:Title       "Semantic Web -- Grundlagen" .
springer:Publisher ex:Name        "Springer Verlag" .
```

likewise triples with coinciding subject and predicate:

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

ex:SemanticWeb ex:Author ex:Hitzler, ex:Krötzsch,
                         ex:Rudolph, ex:Sure ;
               ex:Titel  "Semantic Web -- Grundlagen" .
```

XML Syntax of RDF

- Turtle intuitively understandable, machine-processable
- yet, better tool support and available libraries for XML
- thus: XML syntax more wide-spread

XML Syntax of RDF

- like in XML, name spaces are used in order to disambiguate tag names
- RDF-specific tags have a predefined name space, by convention abbreviated with 'rdf'

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:ex="http://example.org/">

 <rdf:Description rdf:about="http://example.org/SemanticWeb">
  <ex:publishedBy>
   <rdf:Description rdf:about="http://springer.com/Publisher"/>
  </ex:publishedBy>
 </rdf:Description>

</rdf:RDF>
```

XML Syntax of RDF

- the rdf:Description element encodes the subject (the URI of which is stated as the value of the associated rdf:about attribute)
- every element directly nested into an rdf:Description element denotes a predicate (the URI of which is the element name),
- predicate elements in turn contain the triple's object as rdf:Description element

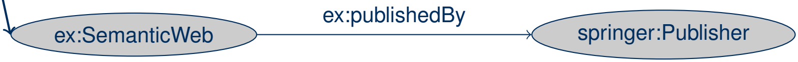
```
:
:
 <rdf:Description rdf:about="http://example.org/SemanticWeb">
  <ex:publishedBy>
   <rdf:Description rdf:about="http://springer.com/Publisher"/>
  </ex:publishedBy>
 </rdf:Description>
:
```

XML Syntax of RDF

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:ex="http://example.org/">
<rdf:Description rdf:about="http://example.org/SemanticWeb">
 <ex:publishedBy>
  <rdf:Description rdf:about="http://springer.com/Publisher"/>
 </ex:publishedBy>
</rdf:Description>
</rdf:RDF>
```

XML Syntax of RDF

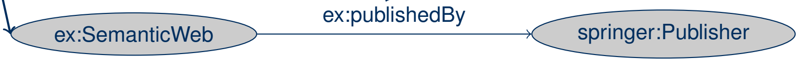
```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:ex="http://example.org/">
<rdf:Description rdf:about="http://example.org/SemanticWeb">
 <ex:publishedBy>
  <rdf:Description rdf:about="http://springer.com/Publisher"/>
 </ex:publishedBy>
</rdf:Description>
</rdf:RDF>
```



TECHNISCHE UNIVERSITÄT DRESDEN

XML Syntax of RDF

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:ex="http://example.org/">
<rdf:Description rdf:about="http://example.org/SemanticWeb">
 <ex:publishedBy>
  <rdf:Description rdf:about="http://springer.com/Publisher"/>
 </ex:publishedBy>
</rdf:Description>
</rdf:RDF>
```



XML Syntax of RDF

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:ex="http://example.org/">
<rdf:Description rdf:about="http://example.org/SemanticWeb">
 <ex:publishedBy>
  <rdf:Description rdf:about="http://springer.com/Publisher"/>
 </ex:publishedBy>
</rdf:Description>
</rdf:RDF>
```

XML Syntax of RDF

- untyped literals can be included as free text into the predicate element
- condensed forms admissible:
  - one subject containing several property elements
  - one object description serves as subject for another triple

```
<rdf:Description rdf:about="http://example.org/SemanticWeb">
 <ex:Title>Semantic Web -- Grundlagen</ex:Title>
 <ex:publishedBy>
  <rdf:Description rdf:about="http://springer.com/Publisher"/>
   <ex:Name>Springer Verlag</ex>Name>
  </rdf:Description>
 </ex:publishedBy>
</rdf:Description>
```

TECHNISCHE UNIVERSITÄT DRESDEN

XML Syntax of RDF

- alternative (but semantically equivalent) representation of literals as XML attributes
- property URIs are then used as attribute names
- object URIs can be given as value of the rdf:resource attribute inside a property tags

```
<rdf:Description rdf:about="http://example.org/SemanticWeb"
              ex:Title="Semantic Web -- Grundlagen">
 <ex:publishedBy rdf:resource="http://springer.com/Publisher"/>
</rdf:Description>
<rdf:Description rdf:about="http://springer.com/Verlag"
              ex:Name="Springer Verlag"/>
```

RDF/XML Syntax: Complications

- name spaces are needed (not just for abbreviation reasons), because colons inside XML elements and attributes are always interpreted as name space delimiters
- problem: in XML, no name spaces in attribute values allowed (would be interpreted as URI schema), thus we cannot write:
  `rdf:about="ex:SemanticWeb"`
- “work around” via XML entities:
  Declaration:
  `<!ENTITY ex 'http://example.org/'>`
  Usage:
  `rdf:resource="&ex;SemanticWeb"`

RDF/XML Syntax: Base URIs

- usage of base URIs:

```
<rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:base="http://example.org/">

 <rdf:Description rdf:about="SemanticWeb">
  <ex:publishedBy rdf:resource="http://springer.com/Publisher"/
 </rdf:Description>

</rdf:RDF>
```

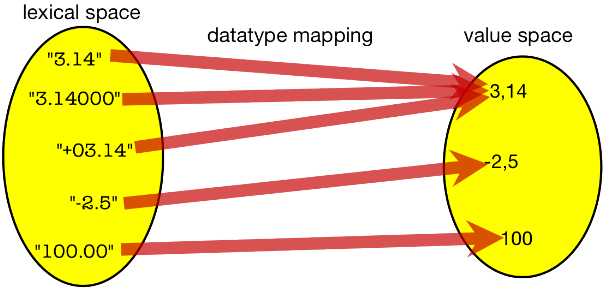
- relative URIs (i.e. those that are to be preceded by the given base URI) are recognized by the absence of a schema part

Agenda

- XML – Motivation
- RDF data model
- Syntax for RDF: Turtle and XML
- Datatypes
- Multi-Valued Relationships
- Blank Nodes
- Lists
- Graph Definitions
- RDF in Practice

Datatypes

Example: `xsd:decimal`



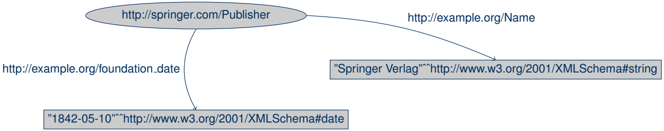
For xsd:decimal holds ”3.14”=”+03.14”
But not for xsd:string !

Datatypes in RDF

- by now: untyped literals, treated like strings (e.g.: ”02”<”100”<”11”<”2”)
- typing allows for a better (more semantic = meaning-adequate) handling of values
- datatypes are themselves denoted by URIs and can essentially be freely chosen
- common: usage of xsd datatypes
- syntax:
  ”datavalue”^^datatype_URI

Datatypes in RDF – Example

Graph:



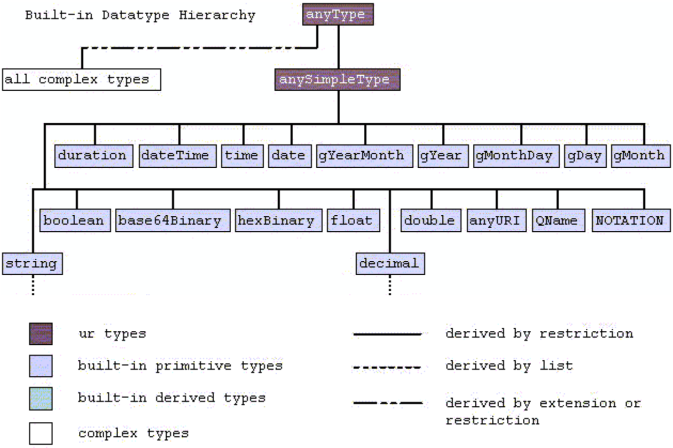
Turtle:

```
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
<http://springer.com/Publisher>
   <http://example.org/Name> "Springer Verlag"^^xsd:string ;
   <http://example.org/foundation\_date> "1842-05-10"^^xsd:date .
```

XML:

```
<rdf:Description rdf:about="http://springer.com/Publisher">
 <ex:Name rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
    Springer Verlag
 </ex:Name>
 <ex:foundation\_date rdf:datatype="http://www.w3.org/2001/XMLSchema#date">
    1842-05-10
 </ex:foundation\_date>
</rdf:Description>
```

XML Schema Datatypes



XML Schema – Facets

- facets are defining properties of a data range
- foundational facet:
  - abstract property for semantically characterizing the values of a value space
  - definition of equality, kind of order relation (total, partial), limits, cardinality, numerical vs. non-numerical
- constraining facet:
  - optional properties to restrict the value space (and thus the lexical space)
  - length (e.g. for strings), minLength, maxLength, pattern (regular expression), enumeration (restriction to explicitly given values), whiteSpace (possible values: preserve, replace (e.g. Tab by Space), collapse (extends replace), maxInclusive, maxExclusive, minExclusive, minInclusive, totalDigits, fractionDigits

XML Schema – duration

- `duration` represents a time span
- six-tuple having as entries Gregorian year, month, day, hour, minute and second, formatted as defined in ISO 8601 §5.5.3.2
- lex. form: PnYnMnDTnHnMnS
- Example: P1Y2M3DT10H30M: duration 1 year, 2 months, 3 days, 10 hours, and 30 minutes)
- admissible facets: pattern, enumeration, whiteSpace, maxInclusive, maxExclusive, minInclusive, minExclusive

XML Schema – dateTime

- `dateTime`: objects with year, month, day, hour and minute given as integer, second given as decimal, optional time zone information
- corresponding decimal value: timeOnTimeline
- lex. form: '-'? yyyy '-' mm '-' dd 'T' hh ':' mm ':' ss ('.' s+)? ((('+' | '-') hh ':' mm) | 'Z')?
- Z stands for UTC (Coordinated Universal Time = Greenwich Mean Time)
- Example: 2002-10-10T12:00:00-05:00: noon of October the 10th 2002, Central Daylight Savings Time/Eastern Standard Time in the US, corresponds to 2002-10-10T17:00:00Z
- admissible facets: pattern, enumeration, whiteSpace, maxInclusive, maxExclusive, minInclusive, minExclusive

XML Schema – time

- `time`: certain point in time, recurring every day
- like `dateTime` but without date
- lex. form: hh ':' mm ':' ss ('.' s+)? ((('+' | '-') hh ':' mm) | 'Z')?
- Example: 12:00:00-05:00: 12:00 Central Daylight Savings Time/Eastern Standard Time in the US, corresponds to 17:00:00 UTC
- admissible facets: pattern, enumeration, whiteSpace, maxInclusive, maxExclusive, minInclusive, minExclusive

XML Schema – date

- `date`: a certain day (interpreted as interval without upper bound)
- like `dateTime` but restricted to date (plus optional time zone information)
- lex. form: '-'? yyyy '-' mm '-' dd ((('+' | '-') hh ':' mm) | 'Z')?
- Example: 2002-10-10-05:00: 10th of October 2002, interval starts -5 hours compared to UTC

XML Schema – gXXX

- `gYearMonth`: a certain Gregorian month in a certain Gregorian year
- `gYear`: a certain Gregorian year
- `gMonthDay`: a (yearly recurring) day of a Gregorian year (like third of April)
- `gDay`: a (monthly recurring) day in the Gregorian calendar (like the third day of each month)
- `gMonth`: a (yearly recurring) month according to the Gregorianischen calendar (erster Monat/Januar)

XML Schema – boolean, base64 und hexBinary

- `boolean`: values of Boolean logic
  - lex. form: { true, false, 1, 0 }
  - admissible facets: pattern, whiteSpace
- `base64`: binary data with base64-encoding with alphabet: a-z, A-Z, 0-9, +, /, = and whitespace
  - admissible facets: length, minLength, maxLength, pattern, enumeration, whiteSpace
- `hexBinary`: binary data with hex encoding: ”0FB7” is hex encoding for 16-bit Integer 4023 (binary representation: 0000.1111.1011.0111)
  - admissible facets: length, minLength, maxLength, pattern, enumeration, whiteSpace

XML Schema – float and double

- `float`: like IEEE single-precision 32-bit floating point type, values $m \times 2^e$ with m, e integers, $|m| < 2^{24}$, $-149 \leq e \leq 104$ plus positive infinity (`INF`) and negative infinity (`-INF`) as well as not-a-number (`NaN`)
- `double`: like IEEE double-precision 64-bit floating point type, values $m \times 2^e$ with m, e integers, $|m| < 2^{53}$, $-1075 \leq e \leq 970$ plus positive infinity (`INF`) and negative infinity (`-INF`) as well as not-a-number

- Examples: -1E4, 1267.43233E12, 12.78e-2, 12 , -0, 0, INF
- not all decimal values within the defined range can be represented
- admissible facets: pattern, enumeration, whiteSpace, maxInclusive, maxExclusive, minInclusive, minExclusive

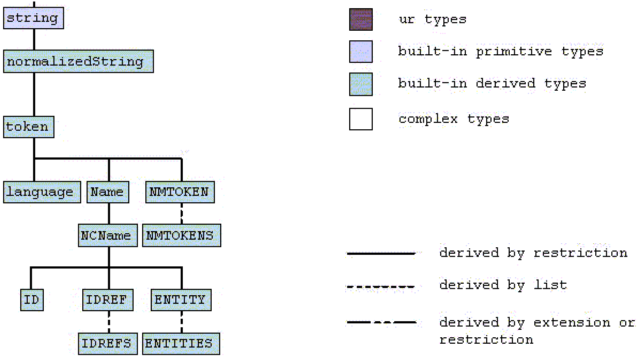
TECHNISCHE UNIVERSITÄT DRESDEN

XML Schema – anyURI, QName, NOTATION

- `anyURI`: a Uniform Resource Identifier, absolut or relative, with or without fragment identifier
- `QName`: a qualified XML Name (name space plus local part, where name space is `anyURI` and local part is `NCName`
- `NOTATION`: like NOTATION attribute type in XML 1.0, cannot be used directly (only derived datatypes)

- admissible facets: length, minLength, maxLength, pattern, enumeration, whiteSpace

XML Schema Datatypes

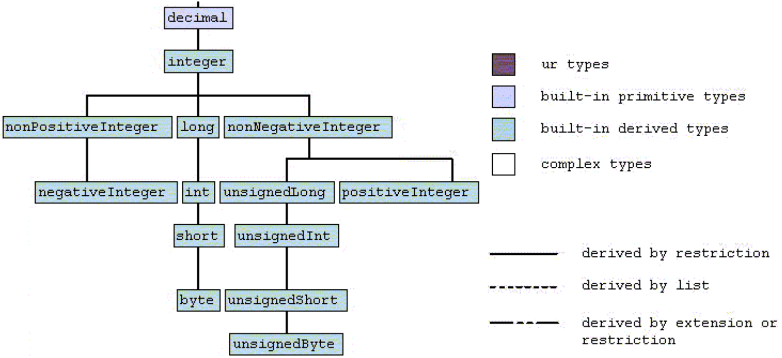


XML Schema – string

- `string`: sequence of characters, where a character is an atomic unit with corresponding code point (integer) in the Universal Character Set

- admissible facets: length, minLength, maxLength, pattern, enumeration, whiteSpace
- further specializations: see specs

XML Schema Datatypes



XML Schema – decimal and integer

- `decimal`: subset of the real numbers, that can be represented as decimal numbers. Value space: numbers representable as $i \times 10^{-n}$ with i, n integers and $n \geq 0$, precision irrelevant: 2.0 equals 2.00
- Examples: -1.23, 12678967.543233, +100000.00, 210
- admissible facets: totalDigits, fractionDigits, pattern, whiteSpace, enumeration, maxInclusive, maxExclusive, minInclusive, minExclusive
- `integer` restriction of `decimal`: `fractionDigits`=0, no decimal point

XML Schema – Types Derived from Integer

- `long` restriction of `integer`: `maxInclusive`=9223372036854775807, `minInclusive`=-9223372036854775808
- `int` restriction of `long`: `maxInclusive`=2147483647, `minInclusive`=-2147483648
- `short` restriction of `int`: `maxInclusive`=32767, `minInclusive`=-32768
- `byte` restriction of `short`: `maxInclusive`=127, `minInclusive`=-128

XML Schema – Types Derived from Integer

- `nonNegativeInteger` restriction of `integer`: `minInclusive`=0
- `positiveInteger` restriction of `nonNegativeInteger`: `minInclusive`=1
- `unsignedLong` restriction of `nonNegativeInteger`: `maxInclusive`=18446744073709551615
- `unsignedInt` restriction of `unsignedLong`: `maxInclusive`=4294967295
- `unsignedShort` restriction of `unsignedInt`: `maxInclusive`=65535
- `unsignedByte` restriction of `unsignedShort`: `maxInclusive`=255

XML Schema – Types Derived from Integer

- `nonPositiveInteger` restriction of `integer`: `maxInclusive`=0
- `negativeInteger` restriction of `nonPositiveInteger`: `minInclusive`=-1

XML Schema – Canonical Values

- there may be several lexical forms for one value
- one of these is picked as the value's canonical form
- useful to detect equivalence between different notations of the same values
- the following lexical forms of the datatype `decimal` represent the same value: 100.5, +100.5, 0100.5, 100.50, 100.500, 100.5000, the canonical variant is: 100.5

TECHNISCHE UNIVERSITÄT DRESDEN

The Predefined Datatype

- rdf:XMLLiteral is the only datatype that is pre-defined within the RDF standard
- denotes arbitrary balanced XML snippets
- in RDF/XML special syntax for unambiguous representation:

```
<rdf:Description rdf:about="http://example.org/SemanticWeb">
 <ex:Titel rdf:parseType="Literal">
    <b>Semantic Web</b><br />
    Grundlagen
 </ex:Titel>
</rdf:Description>
```

Language Information and Datatypes

- language information can only be provided for untyped literals
- Example:

XML:

```
<rdf:Description rdf:about="http://springer.com/Publisher">
 <ex:Name xml:lang="de">Springer Verlag</ex:Name>
 <ex:Name xml:lang="en">Springer Science+Business
                        Media</ex:Name>
</rdf:Description>
```

Turtle:

```
<http://springer.com/Publisher> <http://example.org/Name>
"Springer Verlag"@de, "Springer Science+Business Media"@en .
```

Language Information and Datatypes

According to the spec, the following literals are all different from each other:

```
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>
<http://springer.com/Verlag> <http://example.org/Name>
"Springer Verlag", "Springer Verlag"@de,
"Springer Verlag"^^xsd:string .
```

In practice they are, however, often implemented as equal.

TECHNISCHE UNIVERSITÄT DRESDEN

Agenda

- XML – Motivation
- RDF data model
- Syntax for RDF: Turtle and XML
- Datatypes
- Multi-Valued Relationships
- Blank Nodes
- Lists
- Graph Definitions
- RDF in Practice

Multi-Valued Relationships

- Cooking with RDF:
  "For the preparation of Chutney, you need 1 lb green mango, a teaspoon Cayenne pepper, . . . "
- first modeling attempt:

```
@prefix ex: <http://example.org/> .
ex:Chutney ex:hasIngredient "1lb green mango",
                            "1 tsp. Cayenne pepper",
                            ...
```

- Not satisfactory: ingredients plus amounts encoded as one string. Search for recipes containing green mango not possible (or difficult).

Multi-Valued Relationships

- Cooking with RDF:
  "For the preparation of Chutney, you need 1 lb green mango, a teaspoon Cayenne pepper, . . . "
- second modeling attempt:

```
@prefix ex: <http://example.org/> .
ex:Chutney ex:hasIngredient ex:geenMango;
           ex:amount "1 lb";
           ex:hasIngredient ex:CayennePepper;
           ex:amount "1 tsp.";
           ...
```

- Even worse: no unique assignment of ingredient and amounts possible.

Multi-Valued Relationships

- Problem: we have a proper three-valued (aka: ternary) relationship (cf. databases)

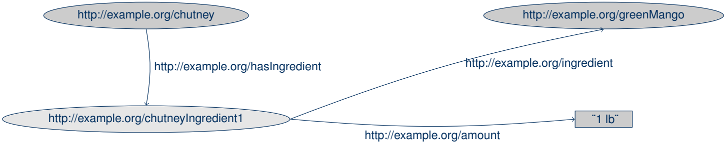
dish	ingredient	amount
chutney	geen mango	1 lb
chutney	Cayenne pepper	1 tsp.

- direct representation in RDF not possible
- solution: introduction of auxiliary nodes

Multi-Valued Relationships

auxiliary nodes in RDF:

- as graph



- Turtle syntax (using rdf:value for the primary component)

```
@prefix ex: <http://example.org/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
ex:chutney             ex:hatZutat ex:chutneyIngredient1 .
ex:chutneyIngredient1  rdf:value   ex:greenMango;
                       ex:amount   "1 lb" .
...
```

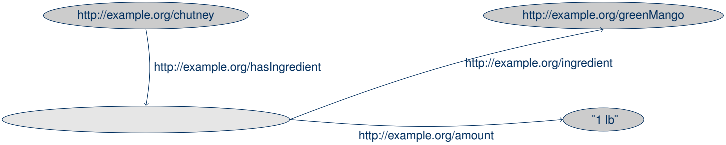
Agenda

- XML – Motivation
- RDF data model
- Syntax for RDF: Turtle and XML
- Datatypes
- Multi-Valued Relationships
- Blank Nodes
- Lists
- Graph Definitions
- RDF in Practice

Blank Nodes

auxiliary nodes in RDF:

- blank nodes (aka bnodes) can be used for resources that need not be named (e.g. auxiliary nodes)
- can be interpreted as existential statement
- syntax (as graph):



Blank Nodes

RDF/XML-Syntax:

```
<rdf:Description rdf:about="http://example.org/chutney">
  <ex:hatZutat rdf:nodeID="id1" />
</rdf:Description>
<rdf:Description rdf:nodeID="id1">
  <ex:ingredient rdf:resource="http://example.org/greenMango" />
  <ex:amount>1 lb<ex:amount/>
</rdf:Description>
```

abbreviated:

```
<rdf:Description rdf:about="http://example.org/chutney">
  <ex:hasIngredient rdf:parseType="Resource">
    <ex:ingredient rdf:resource="http://example.org/greenMango" />
    <ex:amount>1 lb<ex:amount/>
  </ex:hatsIngredient>
</rdf:Description>
```

Blank Nodes

Turtle syntax:

```
@prefix ex: <http://example.org/> .
ex:chutney ex:hasIngredient  _:id1 .
_:id1      ex:ingredient     ex:greenMango ;
           ex:amount         "1 lb" .
```

abbreviated:

```
@prefix ex: <http://example.org/> .
ex:chutney ex:hasIngredient [
           ex:ingredient ex:greenMango ;
           ex:amount     "1 lb" ] .
```

Agenda

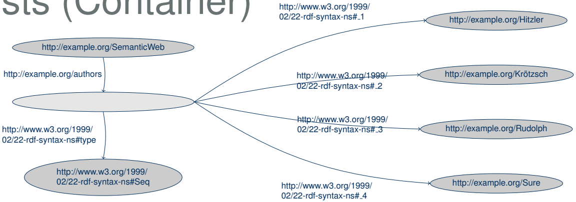
- XML – Motivation
- RDF data model
- Syntax for RDF: Turtle and XML
- Datatypes
- Multi-Valued Relationships
- Blank Nodes
- Lists
- Graph Definitions
- RDF in Practice

Lists

- General data structures for enumerating arbitrary many resources (where order is relevant), e.g. autors of a book
- distinction between
  - open lists (container)
    new entries can be added
  - closed lists (collections)
    new entries can not be added
- These structures are modeled using the already discussed means of representation, i.e. no additional expressivity!

Open Lists (Container)

graph:



abbreviated in RDF/XML:

```
<rdf:Description rdf:about="http://example.org/SemanticWeb">
  <ex:authors>
    <rdf:Seq>
      <rdf:li rdf:resource="http://example.org/Hitzler />
      <rdf:li rdf:resource="http://example.org/Krötzsch />
      <rdf:li rdf:resource="http://example.org/Rudolph />
      <rdf:li rdf:resource="http://example.org/Sure />
    </rdf:Seq>
  </ex:authors>
</rdf:Description>
```

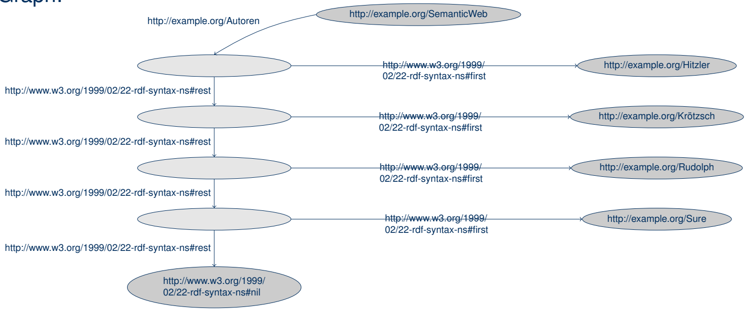
Types of Open Lists

Via `rdf:type` the a list type is assigned to the root node of the list:

- rdf:Seq
  ordered list (sequence)
- rdf:Bag
  unordered set
  indicates that the encoded order is irrelevant
- rdf:Alt
  set of alternatives
  normally only one entry will be relevant

Closed Lists (Collections)

Graph:



underlying idea: recursive deconstruction of the list into head element and (possibly empty) rest list

Closed Lists (Collections)

RDF/XML-Syntax

```
<rdf:Description rdf:about="http://example.org/SemanticWeb>
  <ex:authors rdf:parseType="Collection">
    <rdf:Description rdf:about="http://example.org/Hitzler />
    <rdf:Description rdf:about="http://example.org/Krötzsch />
    <rdf:Description rdf:about="http://example.org/Rudolph />
    <rdf:Description rdf:about="http://example.org/Sure />
  </ex:authors>
</rdf:Description>
```

Turtle

```
@prefix ex: <http://example.org/> .
ex:SemanticWeb ex:authors
  ( ex:Hitzler ex:Krötzsch ex:Rudolph ex:Sure ) .
```

Agenda

- XML – Motivation
- RDF data model
- Syntax for RDF: Turtle and XML
- Datatypes
- Multi-Valued Relationships
- Blank Nodes
- Lists
- Graph Definitions
- RDF in Practice

Graph Definitions

- An RDF triple consists of three components:
  1. the subject, which can be a URI or a bnode,
  2. the predicate, which has to be a URI, and
  3. the object, which can be a URI, a bnode or a Literal.
- The predicate is also denoted as property.
- An RDF graph (or simply graph) is a set of RDF triples. The graph nodes are the subjects and objects of these triples.
- A (proper) subgraph of an RDF graph is a (proper) subset of its triples.
- A ground graph is an RDF graph without bnodes.

Graph Definitions

- A name is a URI reference or a literal.
- A typed literal comprises two names: the literal itself and its type reference (URI)
- A set of names is referred to as a vocabulary.
- The vocabulary of a graph is the set of all names occurring as subject, predicate or object in one of its triples.
- Remark: The URI references which only occur inside the typed literals do not belong to the graph's vocabulary.

TECHNISCHE UNIVERSITÄT DRESDEN

Graph Definitions

- Let M be a mapping from bnodes to a set of literals, bnodes and URI references. We denote M as instance mapping.
- Every graph G' obtained by substituting (some or all) bnodes ℓ in G by $M(\ell)$, is an instance of G.
- An instance with respect to a vocabulary V is an instance in which all names replacing bnodes are from V.
- A proper instance of a graph is an instance wherein at least one bnode has been replaced by a name or identified with another bnode.
- Graphs that only differ in the labels of their bnodes are considered equivalent.

TECHNISCHE UNIVERSITÄT DRESDEN

Graph Definitions

- An RDF graph is lean if it does not have an instance that is a proper subgraph of it.

$\leadsto$ Non-lean graphs are internally redundant.

```
{ex:a ex:p _:x .  _:y ex:p _:x .}        (1)
{ex:a ex:p _:x .  _:x ex:p _:x .}        (2)
```

Graph Definitions

- An RDF graph is lean if it does not have an instance that is a proper subgraph of it.

$\rightsquigarrow$ Non-lean graphs are internally redundant.

```
{ex:a ex:p _:x .  _:y ex:p _:x .}    (1)
{ex:a ex:p _:x .  _:x ex:p _:x .}    (2)
```

- (1) is not lean, but (2) is

Graph Definitions

The merge of two RDF graphs G_1 and G_2 is defined as follows:

- if G_1 and G_2 do not have common bnodes, the merge is the union $G_1 \cup G_2$
- otherwise, the merge of G_1 and G_2 is the union of G_1' und G_2', where G_1' and G_2' are equivalent to G_1 and G_2, respectively, but do not have blank nodes in common
- if this renaming of variables is carried out, one usually says “blank nodes have been are standardized apart”

Agenda

- XML – Motivation
- RDF data model
- Syntax for RDF: Turtle and XML
- Datatypes
- Multi-Valued Relationships
- Blank Nodes
- Lists
- Graph Definitions
- RDF in Practice

Popularty of RDF

- today, a plethora of RDF tools exists
- there are libraries for virtually all programming languges
- freely available systems to work with large RDF data sets (so-called RDF Stores or Triple Stores)
- also commercial players (like Oracle) support RDF
- RDF is basis for other data formats: RSS 1.0, XMP (Adobe), SVG (vector graphics)

Assessment of RDF

- widely supported standard for data storage and interchange
- enables syntax-independent representation of distributed information via a graph-based data model
- pure RDF very oriented toward idividuals
- few possibilities to encode schema knowledge
- $\rightarrow$ RDF Schema (next lecture)

RDFa – RDF-in-attributes

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML+RDFa 1.0//EN"
    "http://www.w3.org/MarkUp/DTD/xhtml-rdfa-1.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
    xmlns:foaf="http://xmlns.com/foaf/0.1/"
    xmlns:dc="http://purl.org/dc/elements/1.1/"
    version="XHTML+RDFa 1.0" xml:lang="en">
  <head>
    <title>John’s Home Page</title>
    <base href="http://example.org/john-d/" />
    <meta property="dc:creator" content="Jonathan Doe" />
    <link rel="foaf:primaryTopic"
        href="http://example.org/john-d/#me" />
  </head>
```

RDFa – RDF-in-attributes

```
  <body about="http://example.org/john-d/#me">
    <h1>John's Home Page</h1>
    <p>My name is <span property="foaf:nick">John D</span>
       and I like <a href="http://www.neubauten.org/"
       rel="foaf:interest" xml:lang="de">Einstürzende
       Neubauten</a>.
    </p>
    <p>
      My <span rel="foaf:interest"
      resource="urn:ISBN:0752820907">favorite book is the
      inspiring <span about="urn:ISBN:0752820907"><cite
      property="dc:title">Weaving the Web</cite> by
      <span property="dc:creator">Tim Berners-Lee</span>
      </span>
     </span>
    </p>
  </body>
</html>
```

RDFa – RDF Version

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:foaf="http://xmlns.com/foaf/0.1/"
    xmlns:dc="http://purl.org/dc/elements/1.1/">
  <rdf:Description rdf:about="http://example.org/john-d/">
    <dc:creator xml:lang="en">Jonathan Doe</dc:creator>
    <foaf:primaryTopic>
      <rdf:Description rdf:about="http://example.org/john-d/#me">
        <foaf:nick xml:lang="en">John D</foaf:nick>
        <foaf:interest rdf:resource="http://www.neubauten.org/"/>
        <foaf:interest>
          <rdf:Description rdf:about="urn:ISBN:0752820907">
            <dc:creator xml:lang="en">Tim Berners-Lee</dc:creator>
            <dc:title xml:lang="en">Weaving the Web</dc:title>
          </rdf:Description>
        </foaf:interest>
      </rdf:Description>
    </foaf:primaryTopic>
  </rdf:Description>
</rdf:RDF>
```

Agenda

- XML – Motivation
- RDF data model
- Syntax for RDF: Turtle and XML
- Datatypes
- Multi-Valued Relationships
- Blank Nodes
- Lists
- Graph Definitions
- RDF in Practice