Semantic Web Technologies I

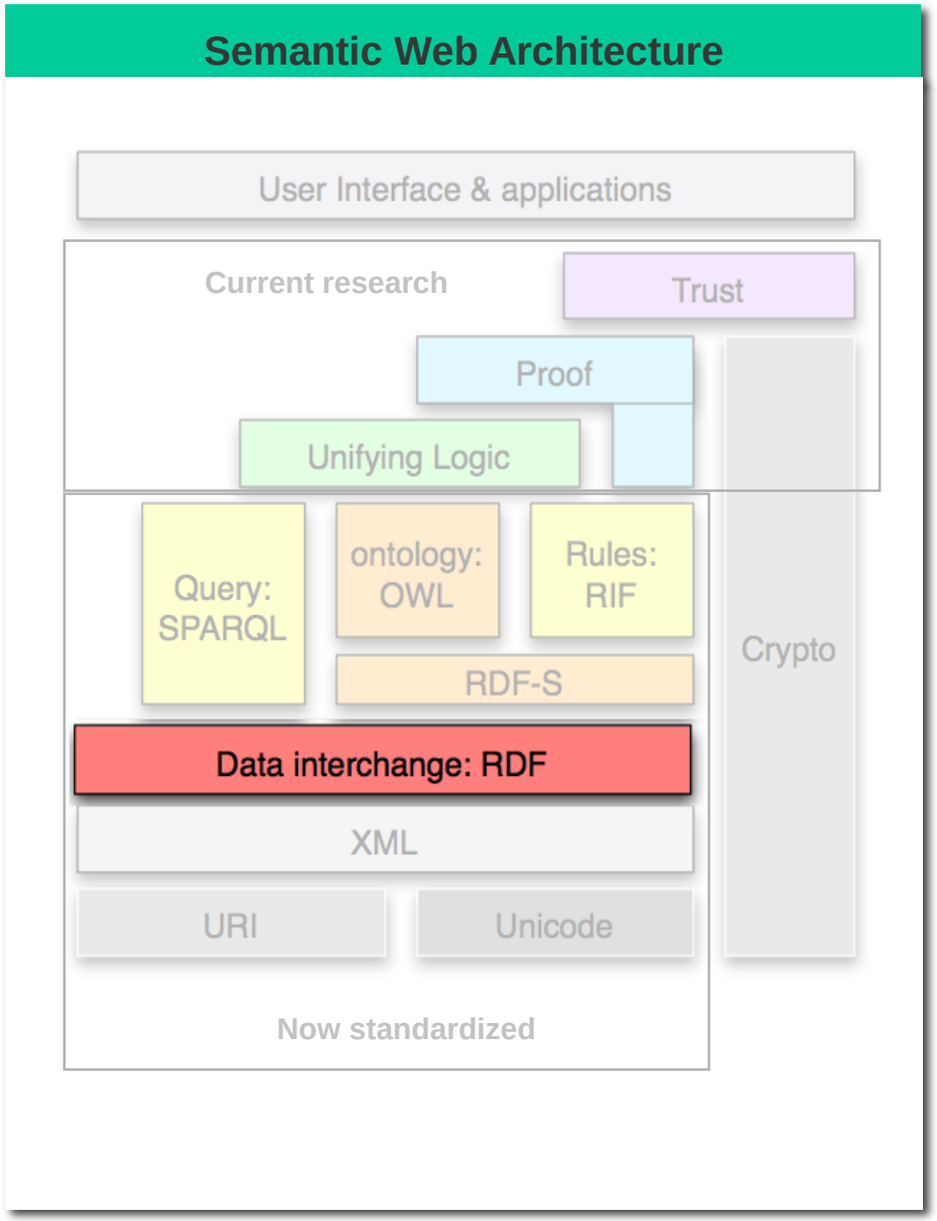
Lehrveranstaltung im WS13/14

Dipl.-Inf. Martin Junghans
Dr. Andreas Harth

www.semantic-web-grundlagen.de

Einführung in RDF

Datum	Thema
23.10.	Einleitung und XML
30.10.	Einführung in RDF
06.11.	RDF Schema
13.11.	Entfällt ggf.
20.11.	Logik Grundlagen
27.11.	Semantik von RDF(S)
04.12.	SPARQL – Syntax und Intuition
11.12.	SPARQL – Semantik
18.12.	OWL – Syntax und Intuition I
08.01.	OWL – Syntax und Intuition II
15.01.	OWL – Semantik und Reasoning
22.01.	Konjunktive Anfragen und Regelsprachen
29.01.	Anwendungen
05.02.	Linked Data
12.02.	Semantische Suche



entspricht ersten Teil von Kapitel 3 des Buches „Semantic Web - Grundlagen“ - siehe auch http://www.semantic-web-grundlagen.de/index.php/Kapitel_3

aifb

Agenda

aifb

- Motivation
- RDF-Datenmodell
- Syntax für RDF: Turtle und XML
- Datentypen
- mehrwertige Beziehungen
- leere Knoten
- Listen

Agenda

aifb

- Motivation
- RDF-Datenmodell
- Syntax für RDF: Turtle und XML
- Datentypen
- mehrwertige Beziehungen
- leere Knoten
- Listen

Unzulänglichkeiten von XML

aifb

- Tag-Namen ambig (durch Namespaces und URIs behebbar)
- Baumstruktur nicht optimal für
  - intuitive Beschreibung der Daten
  - Informationsintegration
- Beispiel: wie kodiert man in einem Baum den Fakt:
  "Das Buch 'Semantic Web - Grundlagen' wird beim Springer-Verlag verlegt"?

Modellierungsprobleme in XML

"Das Buch 'Semantic Web - Grundlagen' wird beim Springer-Verlag verlegt"

```
<Verlegt>
  <Verlag>Springer-Verlag</Verlag>
  <Buch>Semantic Web - Grundlagen</Buch>
</Verlegt>
```

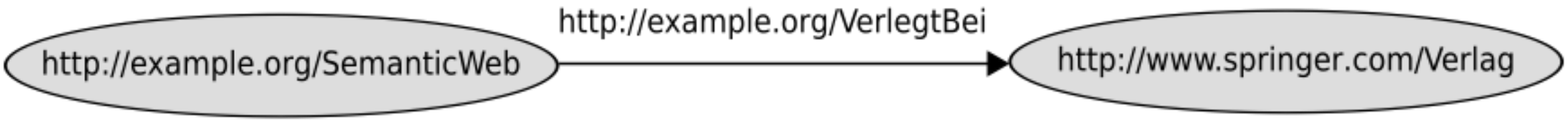
```
<Verlag name="Springer-Verlag">
  <Verlegt buch="Semantic Web - Grundlagen" />
</Verlag>
```

```
<Buch name="Semantic Web - Grundlagen">
  <Verleger verlag="Springer-Verlag" />
</Buch>
```

etc. ....

RDF: Graphen statt Bäume

- Lösung: Darstellung durch (gerichtete Graphen)



Agenda

- Motivation
- RDF-Datenmodell
- Syntax für RDF: Turtle und XML
- Datentypen
- mehrwertige Beziehungen
- leere Knoten
- Listen

Allgemeines zu RDF

aifb

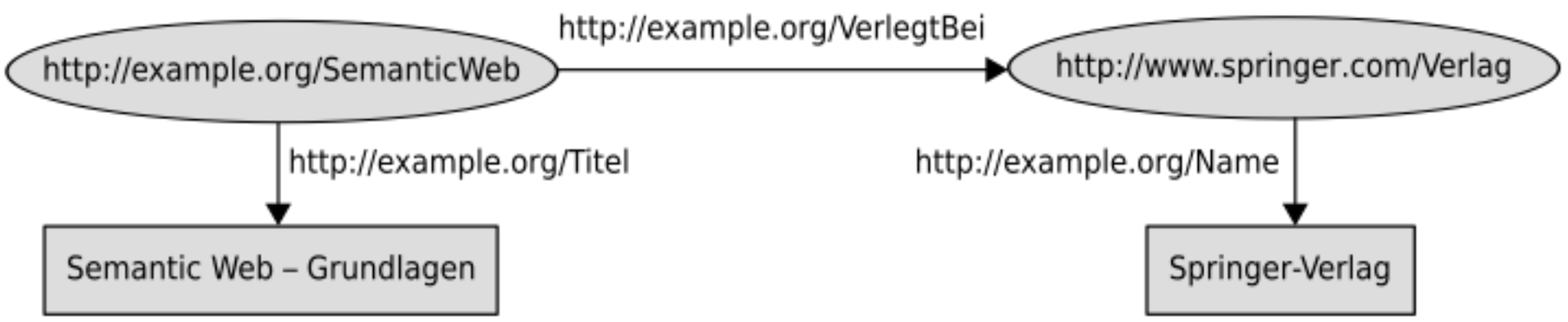
- “Resource Description Framework”
- W3C Recommendation (http://www.w3.org/RDF)
- RDF ist ein Datenmodell
  - ursprünglich: zur Angabe von Metadaten für Web-Ressourcen, später allgemeiner
  - kodiert strukturierte Informationen
  - universelles, maschinenlesbares Austauschformat

Bestandteile von RDF-Graphen

- URIs
  - zur eindeutigen Referenzierung von Ressourcen
  - bereits im Rahmen von XML behandelt
- Literale
  - beschreiben Datenwerte denen keine separate Existenz zukommt
- leere Knoten
  - erlauben Existenzaussagen über ein Individuum mit gewissen Eigenschaften ohne es zu benennen

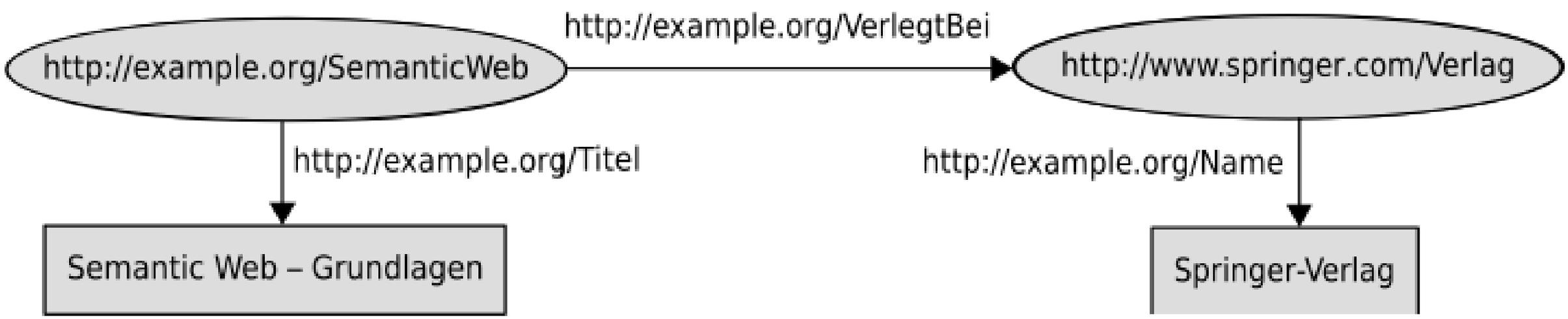
Literale

- zur Repräsentation von Datenwerten
- Darstellung als Zeichenketten
- Interpretation erfolgt durch *Datentyp*
- Literale ohne Datentyp werden wie Zeichenketten behandelt



Graph als Menge von Tripeln

- verschiedene Darstellungsmöglichkeiten für Graphen
- hier verwendet: Liste von (Knoten-Kante-Knoten)-Tripeln



RDF-Tripel

aifb

- Bestandteile eines RDF-Tripels

- angelehnt an linguistische Kategorien, aber nicht immer stimmig

- erlaubte Belegungen:
  Subjekt : URI oder leerer Knoten
  Prädikat: URI (auch Propertys genannt)
  Objekt : URI oder leerer Knoten oder Literal

- Knoten- und Kantenbezeichner eindeutig, daher ursprünglicher Graph aus Tripel-Liste rekonstruierbar

Agenda

aifb

- Motivation
- RDF-Datenmodell
- Syntax für RDF: Turtle und XML
  - Datentypen
  - mehrwertige Beziehungen
  - leere Knoten
  - Listen

Einfache Syntax für RDF

- direkte Auflistung der Tripel:
  - N3: "Notation 3" - umfangreicher Formalismus
  - N-Triples: Teil von N3
  - Turtle: Erweiterung von N-Triples (Abkürzungen)

- Syntax in Turtle:
  - URIs in spitzen Klammern
  - Literale in Anführungszeichen
  - Tripel durch Punkt abgeschlossen
  - Leerzeichen und Zeilenumbrüche außerhalb von Bezeichenern werden ignoriert

Turtle Syntax: Abkürzungen

aifb

- Beispiel

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

- auch in Turtle können Abkürzungen für Präfixe festgelegt werden:

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

ex:SemanticWeb   ex:VerlegtBei  springer:Verlag .
ex:SemanticWeb   ex:Titel       "Semantic Web - Grundlagen" .
springer:Verlag  ex:Name        "Springer-Verlag" .
```

Turtle Syntax: Abkürzungen

- mehrere Tripel mit gleichem Subjekt kann man zusammenfassen:

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

ex:SemanticWeb   ex:VerlegtBei  springer:Verlag ;
                 ex:Titel       "Semantic Web - Grundlagen" .
springer:Verlag  ex:Name        "Springer-Verlag", "Springer" .
```

- ebenso Tripel mit gleichem Subjekt und Prädikat:

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

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

XML-Syntax von RDF

aifb

- Turtle intuitiv gut lesbar und maschinenverarbeitbar
- aber: bessere Tool-Unterstützung und Programmbibliotheken für XML
- daher: XML-Syntax am verbreitetsten

XML-Syntax von RDF

aifb

- wie in XML werden Namensräume eingesetzt, um Tagnamen zu disambiguieren
- RDF-eigene tags haben einen festgelegten Namensraum, der Bezeichner ist standardmäßig ‘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:VerlegtBei>
      <rdf:Description rdf:about="http://springer.com/Verlag">
      </rdf:Description>
    </ex:VerlegtBei>
  </rdf:Description>

</rdf:RDF>
```

XML-Syntax von RDF

- Das `rdf:Description`-Element kodiert das Subjekt (dessen URI wird als Wert des zugehörigen `rdf:about`-Attributs angegeben).
- Jedes geschachtelt im `rdf:Description`-Element enthaltene Element steht für ein Prädikat (dessen URI ist der Elementname), das wiederum das Tripel-Objekt als `rdf:Description`-Element enthält.

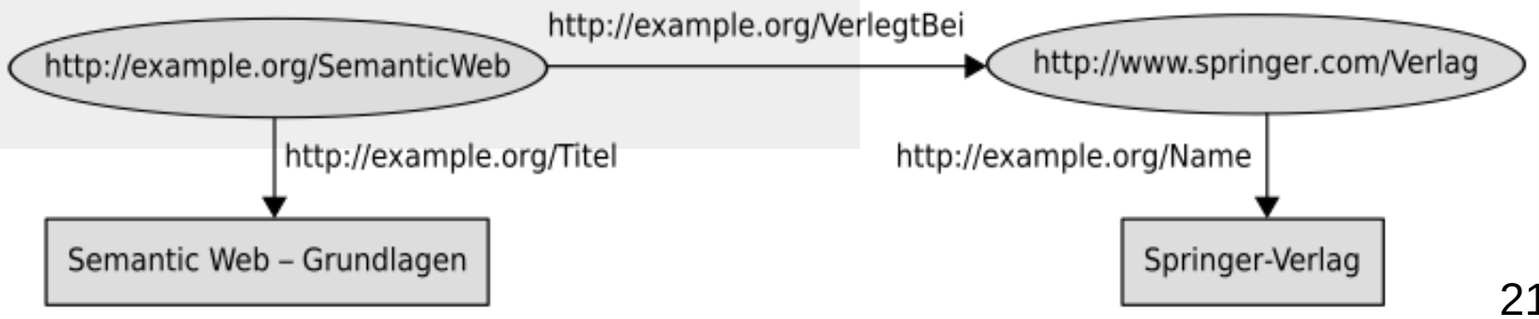
```
<rdf:Description rdf:about="http://example.org/SemanticWeb">
  <ex:VerlegtBei>
    <rdf:Description rdf:about="http://springer.com/Verlag">
    </rdf:Description>
  </ex:VerlegtBei>
</rdf:Description>
```

XML-Syntax von RDF

aifb

- ungetypte Literale können als Freitext in das Prädikatelement eingeschlossen werden
- Verkürzte Darstellung erlaubt:
  - ein Subjekt enthält mehrere Property-Elemente
  - eine Objekt-Description dient als Subjekt für ein weiteres Tripel

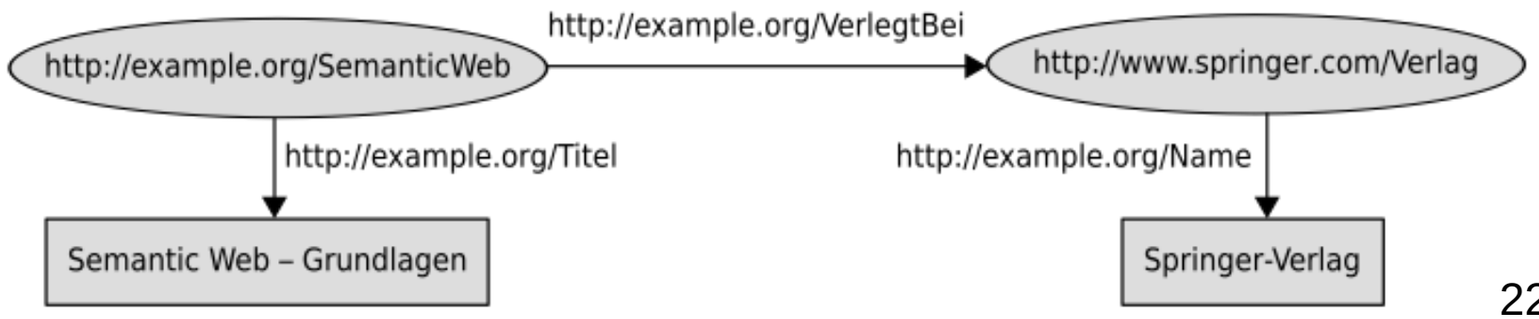
```
<rdf:Description rdf:about="http://example.org/SemanticWeb">
  <ex:Titel>Semantic Web - Grundlagen</ex:Titel>
  <ex:VerlegtBei>
    <rdf:Description rdf:about="http://springer.com/Verlag">
      <ex:Name>Springer-Verlag</ex:Name>
    </rdf:Description>
  </ex:VerlegtBei>
</rdf:Description>
```



XML-Syntax von RDF

- Alternative (aber semantisch gleichwertige) Darstellung für Literale als XML-Attribute
- Attributnamen sind dann die Property-URIs
- Angabe von Objekt-URIs als Wert des `rdf:resource`-Attributs innerhalb eines Property-Tags

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



RDF/XML-Syntax: Komplikationen

- Namensräume sind essentiell (nicht nur Abkürzung), da in XML-Elementen und -Attributen keine Doppelpunkte zulässig, die keine Namensräume kodieren
- Problem: in XML keine Namensräume in Attributwerten möglich (würde im Sinne eines URI-Schemas interpretiert), also z.B. verboten:

```
rdf:about="ex:SemanticWeb"
```

- "Workaround" via XML-Entitäten:
  Deklaration:

```
<!ENTITY ex 'http://example.org/'>
```

  Verwendung:

```
rdf:resource="&ex;SemanticWeb"
```

RDF/XML-Syntax: Basis-URIs

aifb

- Arbeit mit Basis-URIs:

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

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

</rdf:RDF>
```

- Erkennung relativer URIs an Abwesenheit eines Schemateils

Agenda

aifb

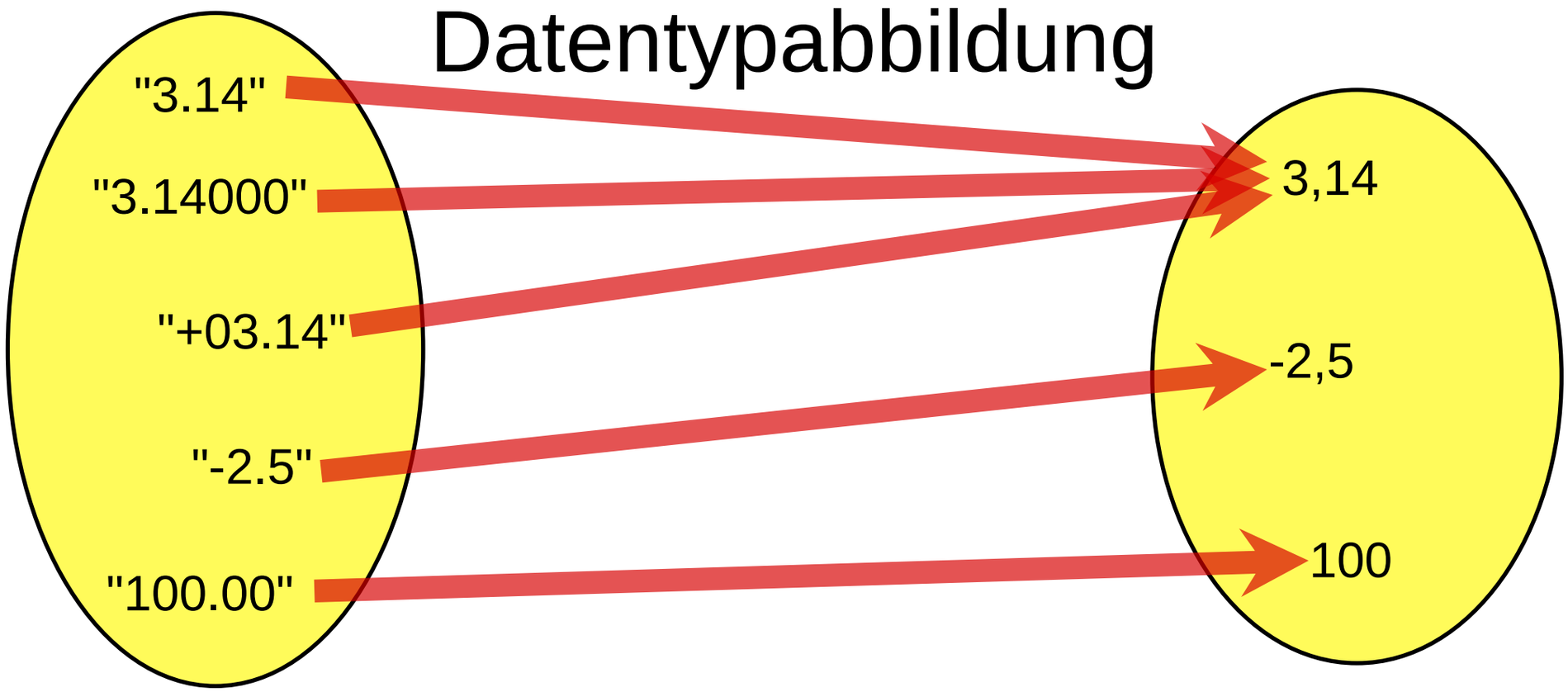
- Motivation
- RDF-Datenmodell
- Syntax für RDF: Turtle und XML
- Datentypen
- mehrwertige Beziehungen
- leere Knoten
- Listen

Datentypen - Abstrakt

aifb

- Beispiel: `xsd:decimal`

lexikalischer Bereich — Wertebereich



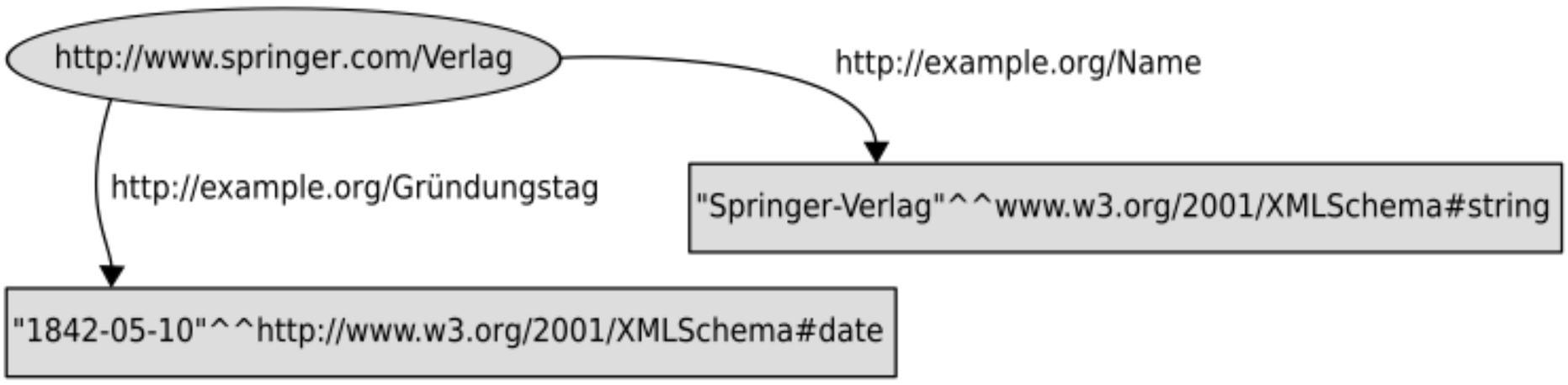
- bzgl. `xsd:decimal` gilt `"3.14"="+03.14"`
  bzgl. `xsd:string` nicht!

Datentypen in RDF

- Bisher: Literale ungetypt, wie Zeichenketten behandelt (also z.B.: "02"<"100"<"11"<"2")
- Typung erlaubt besseren (semantischen = bedeutungsgemäßen) Umgang mit Werten
- Datentypen werden durch URIs identifiziert und sind im Prinzip frei wählbar
- häufig: Verwendung von xsd-Datentypen
- Syntax:
  "*Datenwert*"^^*Datentyp-URI*

Datentypen in RDF - Beispiel

Graph:



- Turtle:

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

- XML:

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

Der vordefinierte Datentyp

- `rdf:XMLLiteral` ist einziger vordefinierter Datentyp in RDF
- bezeichnet beliebige (balancierte) XML-Fragmente
- in RDF/XML besondere Syntax zur eindeutigen Darstellung:

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

Sprachangaben und Datentypen

- Sprachinformationen beeinflussen nur ungetypte Literale
- Beispiel:
  - XML

```
<rdf:Description rdf:about="http://springer.com/Verlag">
   <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/Verlag> <http://example.org/Name>
   "Springer-Verlag"@de, "Springer Science+Business Media"@en .
```

Sprachangaben und Datentypen

- nach RDF-Spezifikation sind demnach die folgenden Literale unterschiedlich:

```
@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 .
```

- ...werden aber häufig (intuitionsgemäß) als gleich implementiert.

Agenda

aifb

- Motivation
- RDF-Datenmodell
- Syntax für RDF: Turtle und XML
- Datentypen
- mehrwertige Beziehungen
- leere Knoten
- Listen

Mehrwertige Beziehungen

aifb

- Kochen mit RDF:
  "*Für die Zubereitung von Chutney benötigt man 450g grüne Mango, einen Teelöffel Cayennepfeffer, ...*"

- erster Modellierungsversuch:

```
@prefix ex: <http://example.org/> .
ex:Chutney  ex:hatZutat   "450g grüne Mango", "1TL Cayennepfeffer"
```

- nicht zufriedenstellend: Zutaten samt Menge als Zeichenkette. Suche nach Rezepten, die grüne Mango beinhalten, so nicht möglich.

Mehrwertige Beziehungen

aifb

- Kochen mit RDF:
  "*Für die Zubereitung von Chutney benötigt man 450g grüne Mango, einen Teelöffel Cayennepfeffer, ...*"

- zweiter Modellierungsversuch:

```
@prefix ex: <http://example.org/> .
ex:Chutney  ex:Zutat    ex:grüneMango;      ex:Menge    "450g" ;
            ex:Zutat    ex:Cayennepfeffer;  ex:Menge    "1TL" .
```

- überhaupt nicht zufriedenstellend: keine eindeutige Zuordnung von konkreter Zutat und Menge mehr möglich.

Mehrwertige Beziehungen

- Problem: es handelt sich um eine echte dreiwertige (auch: ternäre) Beziehung (s. z.B. Datenbanken)

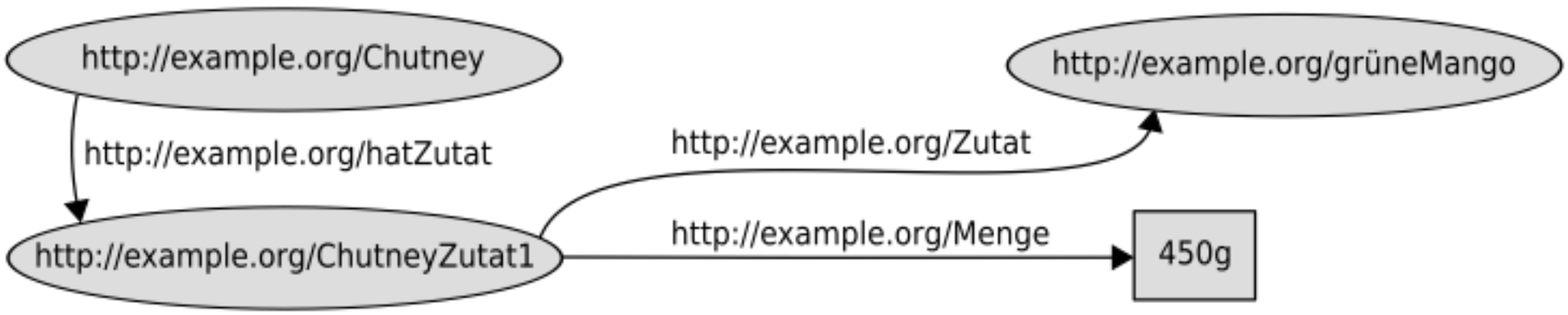
Gericht	Zutat	Menge
Chutney	grüne Mango	450g
Chutney	Cayennepfeffer	1 TL

- direkte Darstellung in RDF nicht möglich
- Lösung: Einführung von Hilfsknoten

Mehrwertige Beziehungen

aifb

- Hilfsknoten in RDF:
  - als Graph



  - Turtle-Syntax (mit Verwendung von `rdf:value`)

```
@prefix ex:  <http://example.org/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
ex:Chutney          ex:hatZutat   ex:ChutneyZutat1 .
ex:ChutneyZutat1    rdf:value     ex:grüneMango;
                    ex:Menge      "450g" .
```

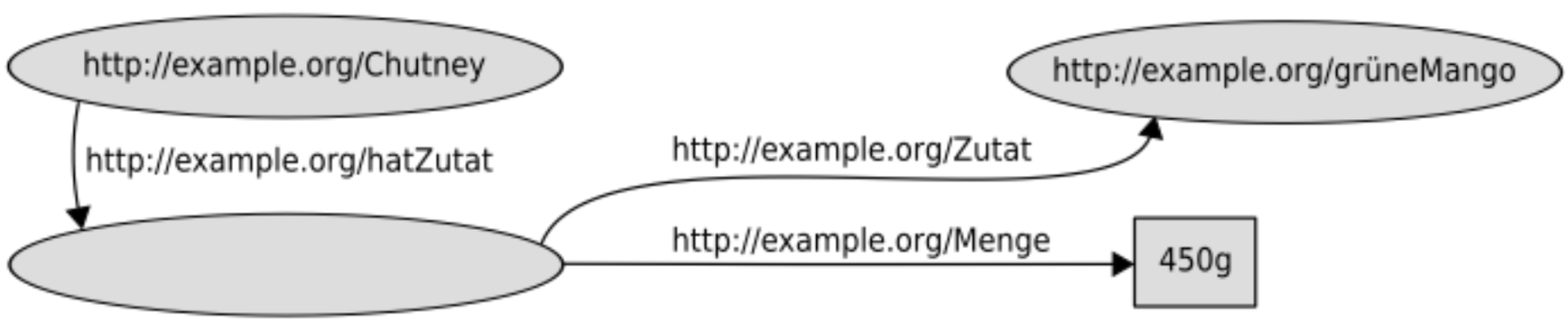
Agenda

aifb

- Motivation
- RDF-Datenmodell
- Syntax für RDF: Turtle und XML
- Datentypen
- mehrwertige Beziehungen
- leere Knoten
- Listen

Leere Knoten

- leere Knoten (blank nodes, bnodes) können für Ressourcen verwendet werden, die nicht benannt werden müssen (z.B. Hilfsknoten)
- können als Existenzaussagen gelesen werden
- Syntax:
  - als Graph



Leere Knoten

- Syntax:
  - RDF/XML-Syntax

```
<rdf:Description rdf:about="http://example.org/Chutney">
   <ex:hatZutat rdf:nodeID="id1" />
</rdf:Description>
<rdf:Description rdf:nodeID="id1">
   <ex:Zutat rdf:resource="http://example.org/grüneMango" />
   <ex:Menge>450g</ex:Menge>
</rdf:Description>
```

  - verkürzt

```
<rdf:Description rdf:about="http://example.org/Chutney">
   <ex:hatZutat rdf:parseType="Resource">
     <ex:Zutat rdf:resource="http://example.org/grüneMango" />
     <ex:Menge>450g</ex:Menge>
   </ex:hatZutat>
</rdf:Description>
```

Leere Knoten

- Syntax:
  - Turtle

```
@prefix ex:   <http://example.org/> .
ex:Chutney     ex:hatZutat   _:id1 .
_:id1          ex:Zutat      ex:grüneMango;   ex:Menge    "450g" .
```

  - verkürzt

```
@prefix ex:   <http://example.org/> .
ex:Chutney     ex:hatZutat
              [ ex:Zutat      ex:grüneMango;   ex:Menge   "450g" ] .
```

Agenda

aifb

- Motivation
- RDF-Datenmodell
- Syntax für RDF: Turtle und XML
- Datentypen
- mehrwertige Beziehungen
- leere Knoten
- Listen

Listen

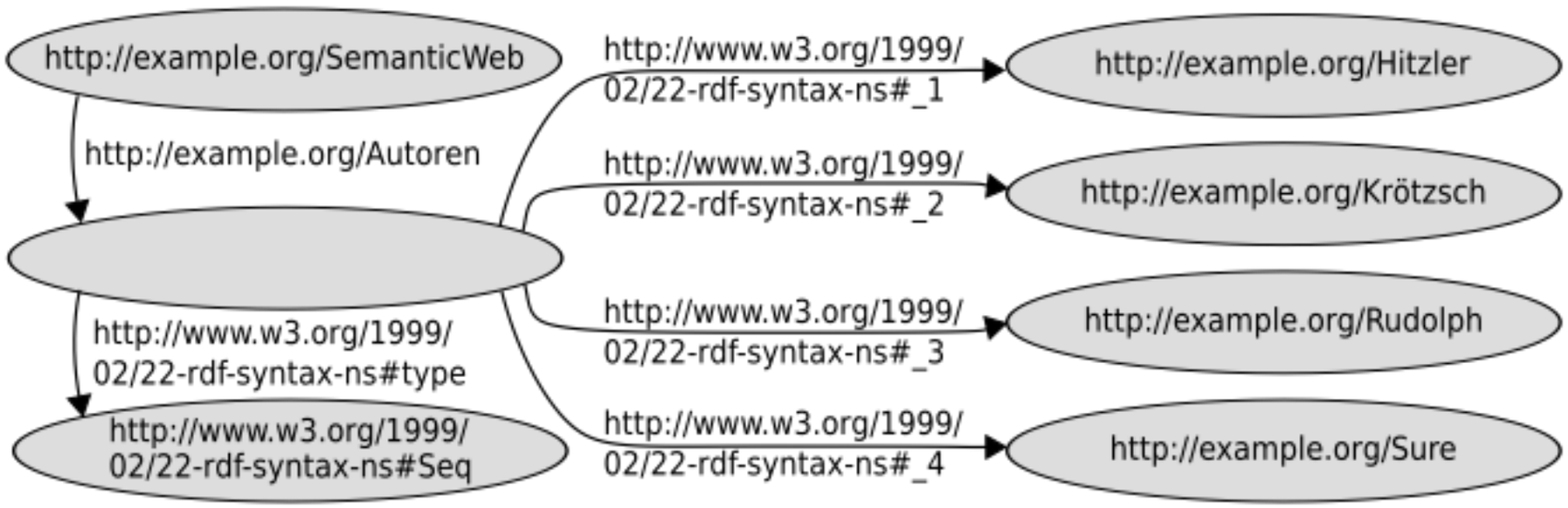
aifb

- allgemeine Datenstrukturen zur Aufzählung von beliebig vielen Ressourcen (Reihenfolge relevant), z.B. Autoren eines Buches
- Unterscheidung zwischen
  - offenen Listen (Container)
    Hinzufügen von neuen Einträgen möglich
  - geschlossenen Listen (Collections)
    Hinzufügen von neuen Einträgen nicht möglich
- Können auch mit bereits vorgestellten Ausdrucksmitteln modelliert werden, also keine zusätzliche Ausdrucksstärke!

Offene Listen (Container)

aifb

- Graph:



- verkürzt in RDF/XML:

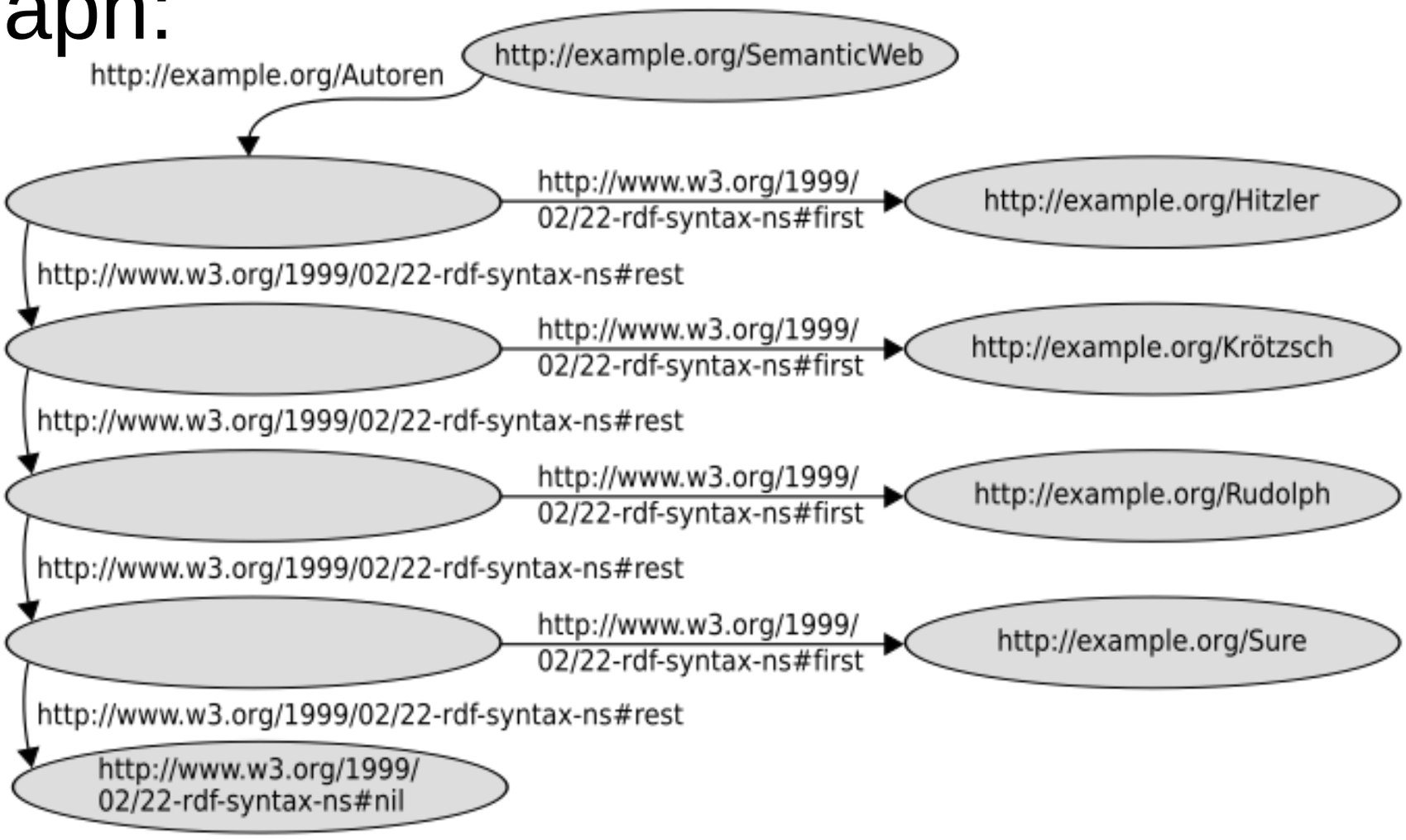
```
<rdf:Description rdf:about="http://example.org/SemanticWeb">
   <ex:Autoren>
      <rdf:Seq>
         <rdf:li rdf:resource="http://www.example.org/Hitzler" />
         <rdf:li rdf:resource="http://www.example.org/Krötzsch" />
         <rdf:li rdf:resource="http://www.example.org/Rudolph" />
    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

Typen offener Listen

- via `rdf:type` wird dem Listen-Wurzelknoten ein Listentyp zugewiesen:
  - `rdf:Seq`
    Interpretation als geordnete Liste (Sequenz)
  - `rdf:Bag`
    Interpretation als ungeordnete Menge
    in RDF kodierte Reihenfolge nicht von Belang
  - `rdf:Alt`
    Menge alternativer Möglichkeiten
    im Regelfall immer nur ein Listeneintrag relevant

Geschlossene Listen (Collections)

- Graph:



- Idee: rekursive Zerlegung der Liste in Kopfelement und (möglicherweise leere) Restliste.

Geschlossene Listen (Collections)

- RDF/XML-Syntax

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

- Turtle

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

Verbreitungsgrad von RDF

- heute existiert Vielzahl von RDF-Tools
- Programmier-Bibliotheken für praktisch jede Programmiersprache
- frei verfügbare Systeme zum Umgang mit großen RDF-Datenmengen (sogenannte RDF Stores oder Triple Stores)
- auch kommerzielle Anbieter (z.B. Oracle) unterstützen zunehmend RDF
- Grundlage für Datenformate: RSS 1.0, XMP (Adobe), SVG (Vektorgrafikformat)

Bewertung von RDF

- weitläufig unterstützter Standard für Speicherung und Austausch von Daten
- ermöglicht weitgehend syntaxunabhängige Darstellung verteilter Informationen in graphbasiertem Datenmodell
- reines RDF sehr "individuenorientiert"
- kaum Möglichkeiten zur Kodierung von Schemawissen
- → RDF Schema (nächste Vorlesung)