hbz

Wissen. Information. Innovation.

Einführung in Linked Open Data

Felix.Ostrowski@hu-berlin.de

Pascal Christoph@hbz-nrw.de

SWIB 2011 Pre-Conference Workshop

Montag, 28. November 2011

Hamburg

Ablauf

- Einleitung: Daten – Graphen – Tripel
- Gruppenarbeit
- URIs und Namensräume
- Gruppenarbeit
- Pause
- Open Data Prinzipien
- Gruppenarbeit
- Bezeichnung & Beschreibung
- Content-Negotiation
- Gruppenarbeit
- Pause
- Triple Stores & SPARQL
- Gruppenarbeit
- Pause
- RDF Schema
- Gruppenarbeit
- Resümee, Fragen & Diskussion

hbz

Linked Open Data

- Es geht um **Daten** …
- … genauer: um **offene** Daten …
- … noch genauer: um **verknüpfte** offene Daten!

hbz

Daten, wie wir sie kennen


```
LDR             ------M2.01200024------h
FMT             MH
001             |a HT016905880
002a            |a 20110726
003             |a 20110729
026             |a HBZHT016905880
030             a|1uc||||||17
036a            |a NL
037b            |a eng
050             a|||||||||||||
051             m|||f|||
070             |a 294/61
070b            |a 361
080             |a 60
100             |a Allemang, Dean |9 136636187
104a            |a Hendler, James A. |9 115664564
331             |a Semantic web for the working ontologist
335             |a effective modeling in RDFS and OWL
359             |a Dean Allemang ; Jim Hendler
403             |a 2. ed.
410             |a Amsterdam [u.a.]
412             |a Elsevier MK
425a            |a 2011
433             |a XIII, 354 S. : graph. Darst.
540a            |a 978-0-12-385965-5
```

(Wenn man ehrlich ist, sind wir die einzigen die solche Daten kennen, und es gibt nicht all zu viele Dinge, die man so beschreibt. Außerdem braucht man Fachwissen, um „Links“ überhaupt identifizieren zu können.)

hbz

Daten, wie andere sie kennen

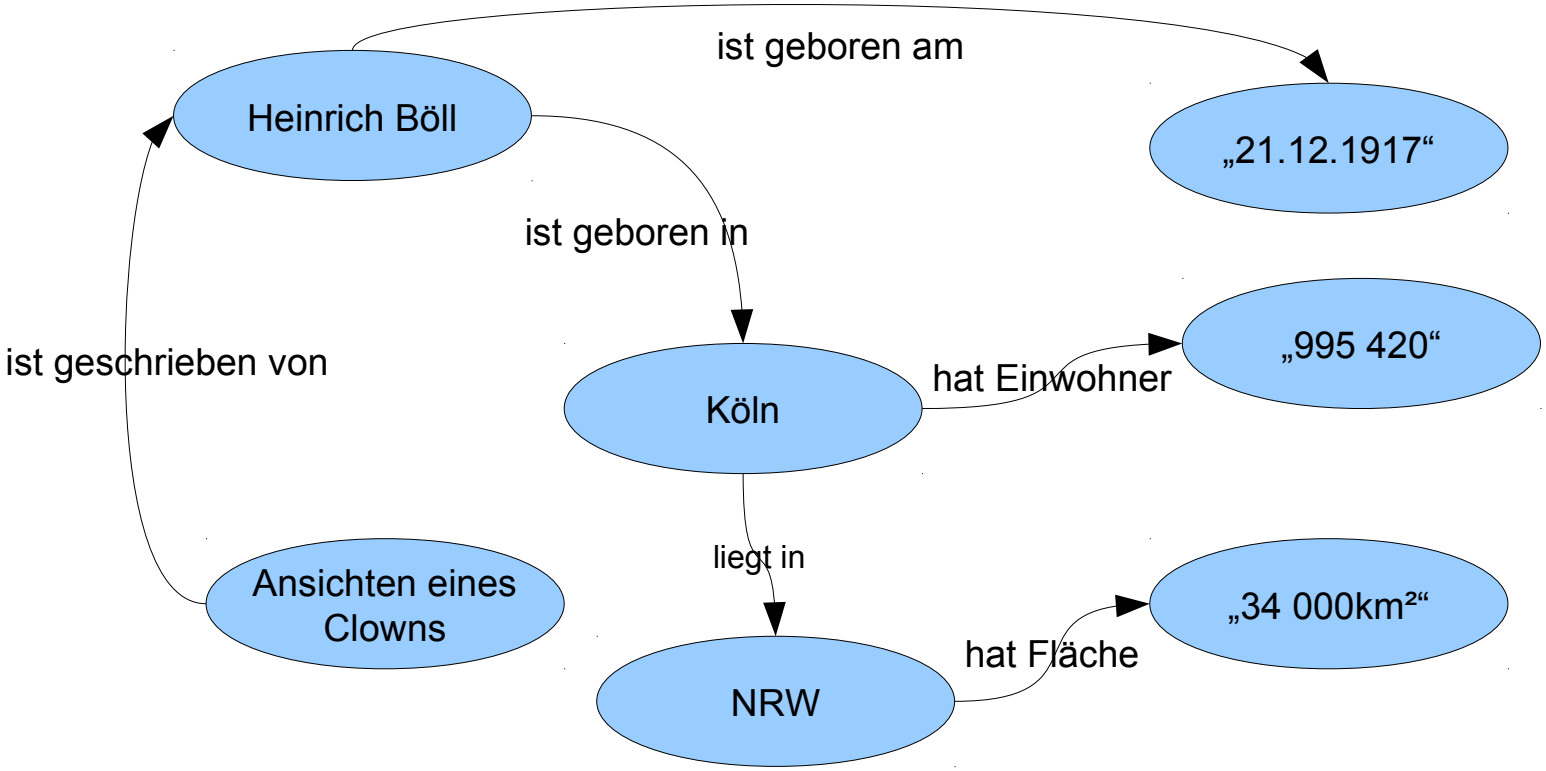
```
+-----------+-----------+----------+----------+
| id        | firstname | lastname | birthday |
+-----------+-----------+----------+----------+
| 136636187 | Dean      | Allemang | NULL     |
+-----------+-----------+----------+----------+

+-------------+-----------------------------------------+-----------+
| id          | title                                   | author    |
+-------------+-----------------------------------------+-----------+
| HT016905880 | Semantic web for the working ontologist | 136636187 |
+-------------+-----------------------------------------+-----------+
```

```
<book id="HT016905880">
  <title>Semantic web … </title>
  <author id="136636187">
    <firstname>Dean</firstname>
    <lastname>Allemang</lastname>
  </author>
</book>
```

(Um auch hier ehrlich zu sein: „andere“ heißt nicht „jeder“, aber immerhin kann man damit schon einige Dinge mehr Beschreiben. Eigentlich sogar fast alles. Das „Link-Problem“ allerdings bleibt.)

hbz

Daten, wie das Netz sie mag



(Sieht ja auch aus wie ein Netz oder, wenn man so will, ein gerichteter benannter **Graph**.)

hbz

Ein Computer kann mit solchen Diagrammen natürlich nichts anfangen. Das **Graphenmodell** ist ein **abstraktes**, aber wir können es für den Computer konkretisieren.

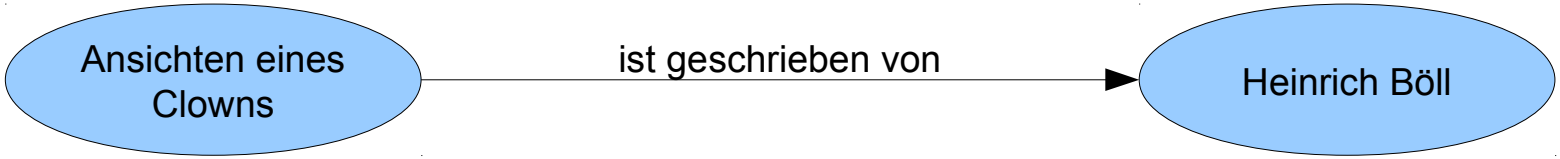
Graphen, (fast) wie Computer sie mögen

```
<Ansichten eines Clowns> <ist geschrieben von> <Heinrich Böll> .
<Heinrich Böll> <hat Vornamen> „Heinrich“ .
<Heinrich Böll> <hat Nachnamen> „Böll“ .
<Heinrich Böll> <ist geboren am> „21.12.1917“ .
<Heinrich Böll> <ist geboren in> <Köln> .
<Köln> <liegt in> <NRW> .
<Köln> <hat Einwohner> „995420“ .
<Köln> <hat Fläche> „34000km²“ .
```

(Diese Schreibweise nennt sich Turtle und ist eine von mehreren Schreibweisen des Datenmodells **RDF**. RDF steht für „**Resource Description Framework**“ und ist der de-facto Standard zur Publikation von Linked-Data.
Vorteil von Turtle: man kann es auch als Mensch einigermaßen gut lesen.)

hbz

Grundbaustein: das Tripel




```
<Ansichten eines Clowns> <ist geschrieben von> <Heinrich Böll> .
```

(Ein Tripel ist der kleinstmögliche Graph. Man spricht bei seinen Bestandteilen auch von **Subjekt**, **Prädikat** und **Objekt**.)

hbz

Sie sind dran!

Legen Sie im Etherpad ein Dokument für Ihre Gruppe an, und beschreiben sie darin die Mitglieder. Die Prädikate können Sie sich zunächst einfach ausdenken.

(Sollte Ihnen wohler dabei sein, können Sie auch Kunstfiguren erfinden.)

Wofür steht …

➔… <Heinrich Böll>,

➔… <Köln> und

➔… <NRW>;

was bedeutet

➔<hat Vornamen>,

➔<liegt in> und

➔<hat Einwohner>?

(Und wie kann es sein, dass Peter Müller in zwei verschiedenen Städten geboren ist?)

Wir brauchen **eindeutige Referenzierung**!

Normdaten sind ein guter Anfang,
aber die verstehen wieder nur wir.
Im Netz verwendet man **URI**s!

(URI steht für **Uniform Resource Identifier**)

hbz

URI

=

scheme ":" hier-part ["?" query] ["#" fragment]

(???)

hbz

http://de.wikipedia.org/wiki/Uniform_Resource_Identifier

ftp://ftp.is.co.za/rfc/rfc3986.txt

file:///home/fo/doc/swib11/slides.odp

urn:isbn:978-1608454303

(Für HTTP-URIs gilt: es muss nicht immer einen Domänennamen geben. IP-Adressen funktionieren auch: http://192.168.0.124)

hbz

Graphen, wie Computer sie wirklich mögen

```
<urn:isbn:978-3423004008> <http://purl.org/dc/terms/creator> <http://d-nb.info/gnd/118512676> .
<http://d-nb.info/gnd/118512676> <http://xmlns.com/foaf/0.1/givenName> „Heinrich“ .
<http://d-nb.info/gnd/118512676> <http://xmlns.com/foaf/0.1/familyName> „Böll“ .
<http://d-nb.info/gnd/118512676> <http://xmlns.com/foaf/0.1/birthday> „21.12.1917“ .
```

(Ein angenehmer Nebeneffekt bei der Verwendung von HTTP-URIs, auf dem Linked Data basiert, ist ihre **Dereferenzierbarkeit**. Folgt man einem solchen Link, so bekommt man eine **Beschreibung** der Resource bzw. des **Vokabulars** aus dem das Prädikat stammt. Dazu später mehr.)

hbz

Graphen, für Computer und Menschen lesbar

```
@prefix dc:   <http://purl.org/dc/terms/> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix gnd:  <http://d-nb.info/gnd/> .

<urn:isbn:978-3423004008> dc:creator gnd:118512676 .
gnd:118512676 foaf:givenName „Heinrich“ .
gnd:118512676 foaf:familyName „Böll“ .
gnd:118512676 foaf:birthday „21.12.1917“ .
```

(Bei den durch Präfixe abgekürzten URIs spricht man auch von den **Namensräumen**, in denen Resourcen und Prädikate existieren.)

hbz

Aber da fehlt doch was, was ist mit den restlichen Daten?

```
<http://d-nb.info/gnd/118512676> <ist geboren in> <Köln> .
<Köln> <liegt in> <NRW> .
<Köln> <hat Einwohner> „995420“ .
<Köln> <hat Fläche> „34000km²“ .
```

(Nicht alles hat bereits einen URI, weder Entitäten noch Prädikate. Auf der Suche nach URIs für Entitäten ist die DBpedia eine gute erste Anlaufstelle. Nicht so gut sieht es bislang bei Vokabularen aus, Schemapedia ist aber einen Versuch wert.)

hbz

Noch ein paar Tricks

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

:ostrowski foaf:givenName „Felix“ .
:ostrowski foaf:familyName „Ostrowski“ .
:ostrowski foaf:birthday „28.05.1981“ .

<> dc:creator :ostrowski .
```

(Wenn etwas noch keinen URI hat, verwenden wir Identifikatoren, die relativ zur URL des beschreibenden Dokumentes sind. Da nicht zwei Dokumente an der selben Stelle liegen können, müssen diese Identifikatoren nur innerhalb des Dokumentes eindeutig sein. „<>“ ist eine Kurzschreibweise für das aktuelle Dokument.)

hbz

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

:ostrowski foaf:givenName „Felix“ ;
           foaf:familyName „Ostrowski“ ;
           foaf:birthday „28.05.1981“ ;
           foaf:based_near dbpedia:Berlin ,
                           dbpedia:Berlin-Kreuzberg .
```

(Folgen mehrere Aussagen über die selbe Resource, muss diese nicht wiederholt werden. Die Prädikate werden durch ein Semikolon getrennt. Folgen mehrere Werte für das selbe Prädikat einer Resource, werden die Objekte durch Kommas getrennt aufgeführt.)

```
@prefix dbpedia: <http://de.dbpedia.org/resource/> .
@prefix skos:    <http://www.w3.org/2004/02/skos/core#> .

dbpedia:Köln skos:prefLabel „Köln“@de ,
                            „Cologne“@en ,
                            „Colonia“@it .
```

(Literale können mit Sprachangaben versehen werden. Ob Sie gültiges Turtle produzieren, können Sie z.B. hier überprüfen.)

hbz

Sie sind dran!

Einigen Sie sich in Ihrer Gruppe auf Identifikatoren und beschreiben Sie die Mitglieder anhand des FOAF Vokabulars. Geben Sie dabei auch an, dass sie sich gegenseitig kennen! Nutzen Sie darüber hinaus DC Terms, um sich als Autoren mit dem Dokument zu verknüpfen (und wenn Sie möchten auch für weitere Metadaten des Dokumentes).

hbz

Pause

Exkurs: **Open** Data

Sie sind dran!

Einigen Sie sich in ihrer Gruppe auf eine Creative Commons Lizenz für ihr Dokument und verknüpfen Sie ihr Dokument mit dieser!

(Es bietet sich das Prädikat <http://creativecommons.org/ns#license> hierfür an, aber eine Suche im Netz offenbart weitere Alternativen.)

hbz

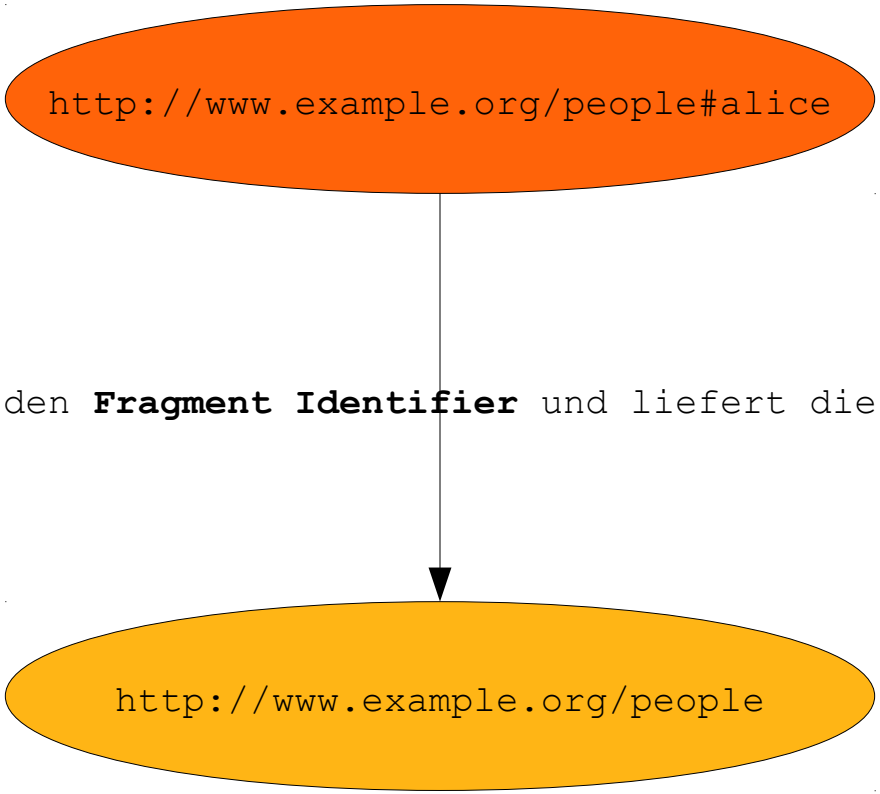
Linked Data in Action

hbz

Bezeichnung und **Beschreibung** sind zu unterscheiden! Im Linked-Data-Paradigma gibt es zwei gängige Möglichkeiten, diese zu verknüpfen.

hbz

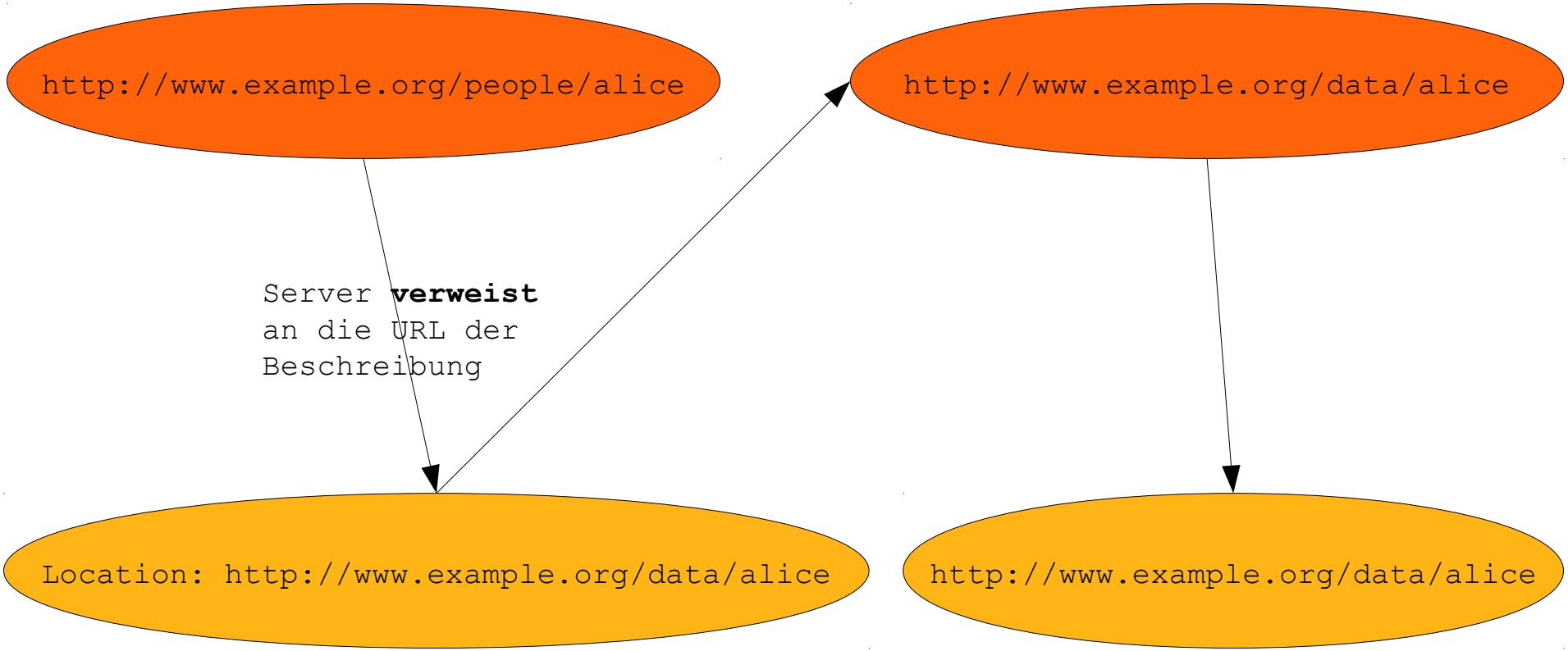
Hash URIs



Server ignoriert den **Fragment Identifier** und liefert die Beschreibung aus

hbz

HTTP 303 Redirects

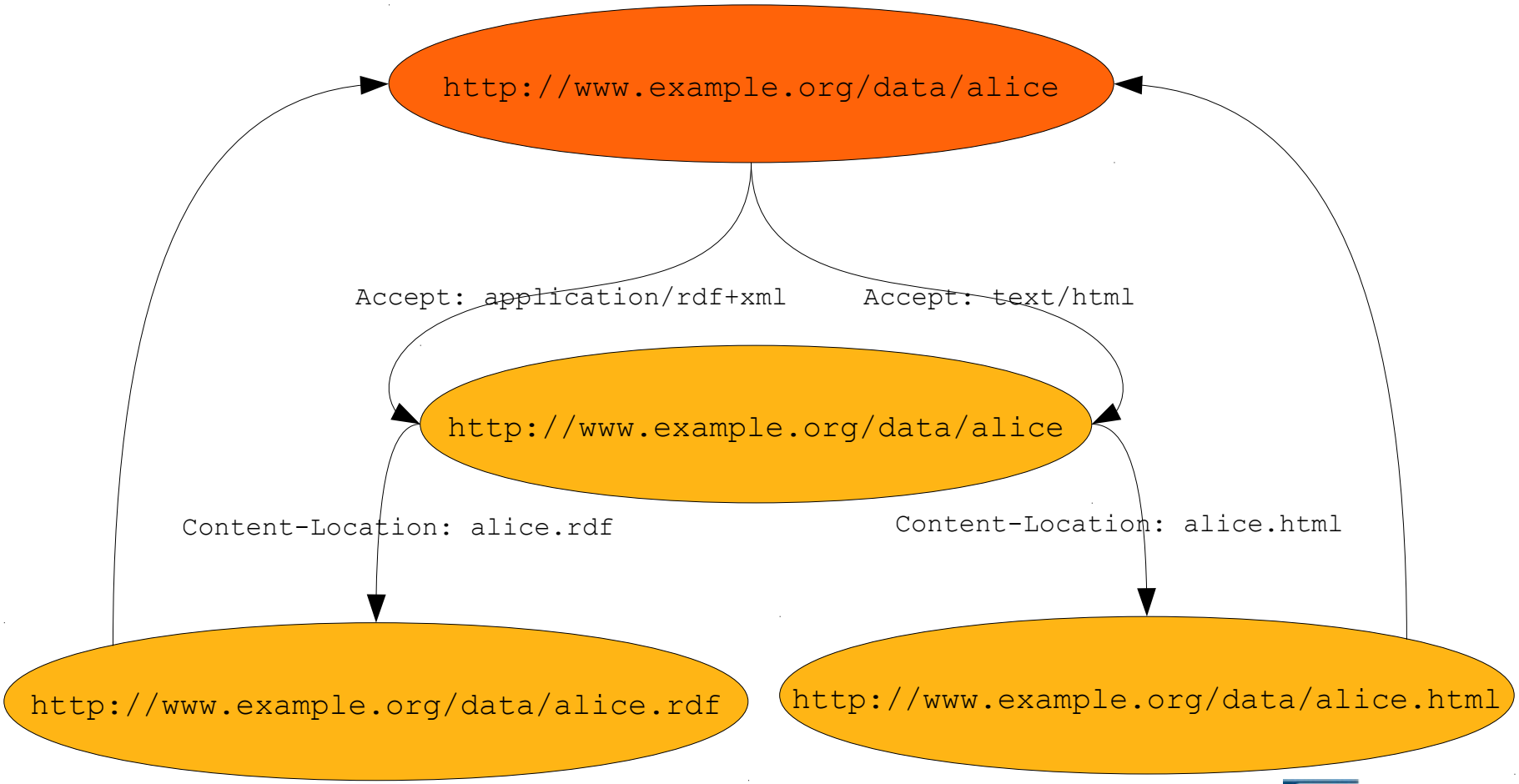


hbz

Die Beschreibung einer Ressource kann in unterschiedlichen **Formaten** verfügbar gemacht werden. Welches Format ausgeliefert wird, kann per **Content-Negotiation** ausgehandelt werden.

hbz

Verhandlungssache



hbz

Der einfachste Weg zur Publikation von Linked Data ist von **statischen RDF-Dateien** in denen **Hash-URIs** verwendet werden auf einem Webserver mit aktivierten „**Multiviews**“ für die Content-Negotiation.

hbz

```
# /var/www/personen.ttl

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

:ostrowski
    foaf:givenName "Felix" ;
    foaf:familyName "Ostrowski" .

:christoph
    foaf:givenName "Pascal" ;
    foaf:familyName "Christoph" .
```

```
# /var/www/.htaccess
AddType text/turtle .ttl
AddType application/rdf+xml .rdf
Options +MultiViews
```

Mit diesem einfachen Setup lauten die **URIs der Personen:**
http://localhost/personen#ostrowski
http://localhost/personen#christoph
Die **URL der Beschreibung** ist:
http://localhost/personen
Content-Negotiation für Turtle und RDF/XML ist aktiviert.

```
# /var/www/personen.rdf

<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF
   xmlns:foaf="http://xmlns.com/foaf/0.1/"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns="http://localhost/lodws/test#">
  <rdf:Description rdf:about="http://localhost/lodws/test#christoph">
    <foaf:familyName>Christoph</foaf:familyName>
    <foaf:givenName>Pascal</foaf:givenName>
  </rdf:Description>
  <rdf:Description rdf:about="http://localhost/lodws/test#ostrowski">
    <foaf:familyName>Ostrowski</foaf:familyName>
    <foaf:givenName>Felix</foaf:givenName>
  </rdf:Description>
</rdf:RDF>
```

hbz

Demo

Sie sind dran!

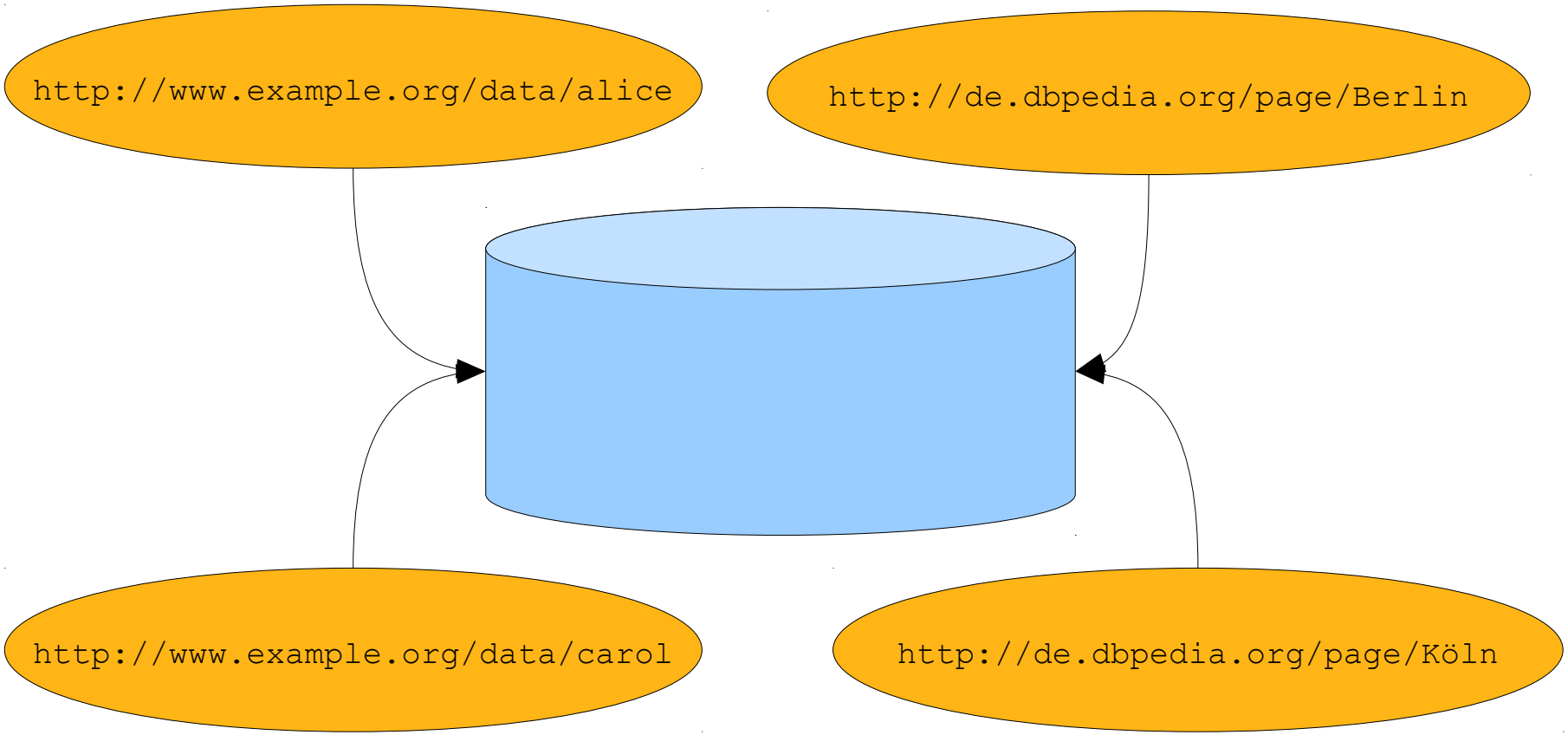
Verknüpfen Sie sich in Ihrer Beschreibung mit Personen aus anderen Gruppen, die Sie kennen. Dies muss nicht reziprok geschehen.

Verlinken Sie außerdem auf ihren (ungefähren) Wohnort. Verwenden Sie dazu die DBpedia.

Pause

Einzelne maschinenlesbare Beschreibungen sind nützlich, aber da geht noch mehr! RDF ist ein **verteiltes** Datenmodell, so dass mehrere Beschreibungen einfach **zusammengeführt** werden können. Außerdem existieren spezielle **Datenbanken**, die **Abfragen** über RDF-Daten ermöglichen.

Triple Stores



hbz

SPARQL ermöglicht Abfragen über die Daten eines Triple Stores. Grundlage sind einfache „Graph-**Schablonen**“. Diese sehen fast so aus wie einfache **Tripel**, mit dem Unterschied, dass sie **Variablen** enthalten.

```
@prefix    ex: <http://example.org/people#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .

ex:alice foaf:name „Alice“ .
```

```
PREFIX    ex: <http://example.org/people#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

SELECT * WHERE {
  ex:alice foaf:name ?name .
}
```

name
„Alice“

hbz

```
@prefix   ex: <http://example.org/people#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .

ex:alice foaf:name „Alice“ ;
         foaf:knows ex:bob .
ex:bob   foaf:name „Bob“ ;
         foaf:knows ex:carol .
ex:carol foaf:name „Carol“ ;
         foaf:knows ex:alice .
```

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

SELECT ?name1 ?name2 WHERE {
    ?person1 foaf:knows ?person2 .
    ?person1 foaf:name ?name1 .
    ?person2 foaf:name ?name2 .
}
```

name1	name2
„Alice“	„Bob“
„Bob“	„Carol“
„Carol“	„Alice“

hbz

```
@prefix      ex: <http://example.org/people#> .
@prefix    foaf: <http://xmlns.com/foaf/0.1/> .
@prefix dbpedia: <http://de.dbpedia.org/resource/> .

ex:alice foaf:name „Alice“ ;
         foaf:knows ex:bob ;
         foaf:based_near dbpedia:Berlin .
ex:bob   foaf:name „Bob“ ;
         foaf:knows ex:carol ;
         foaf:based_near dbpedia:Dresden .
```

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT ?name ?ortname WHERE {
    ?person1 foaf:knows ?person2 .
    ?person2 foaf:name ?name .
    ?person2 foaf:based_near ?ort .
    ?ort rdfs:label ?ortname .
}
```

name	ortname
„Bob“	„Dresden“@de

hbz

Sie sind dran!

Nutzen Sie SPARQL, um ihren „Bekanntenkreis“ zu analysieren. Ermitteln Sie zum Beispiel, wen Sie über ein oder zwei Ecken kennen und wer ebenfalls aus ihrer Stadt kommt.

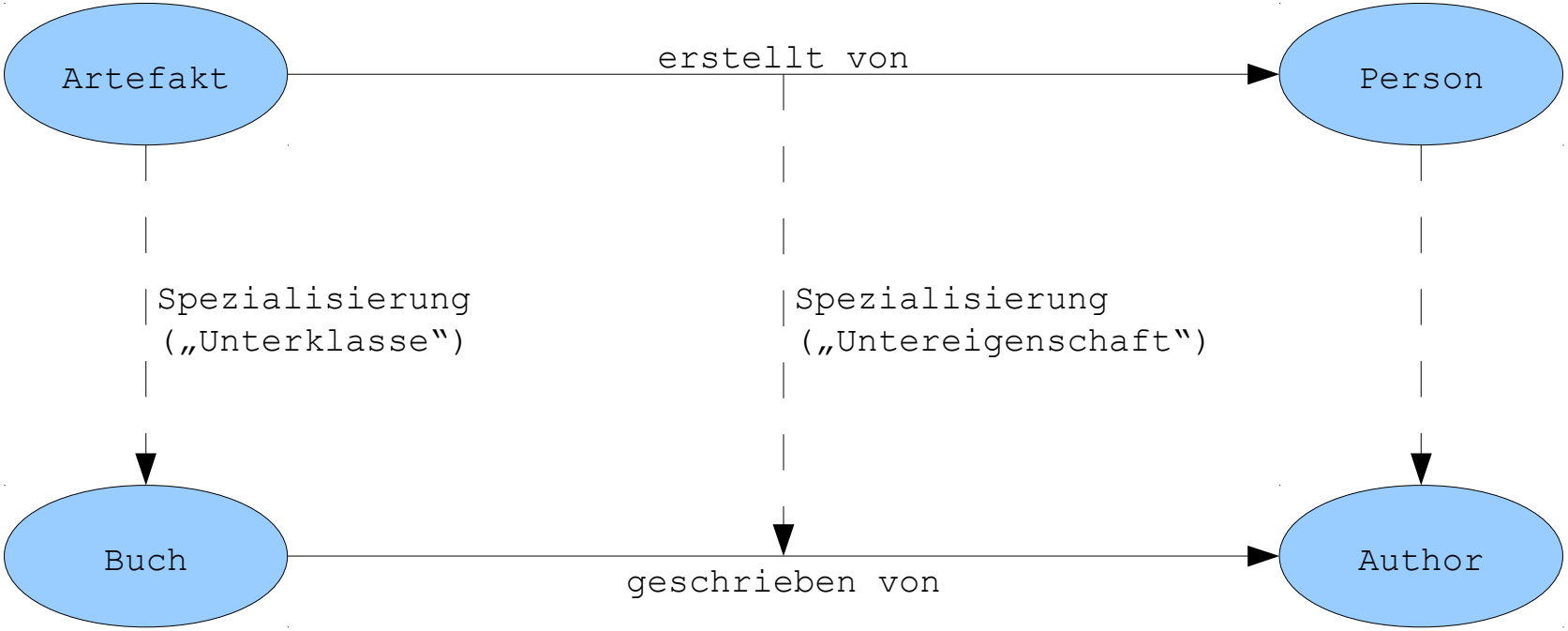
Pause

Jetzt kommt die **Semantic** ins **Web**

Die **Klassen** und **Eigenschaften**, die verwendet werden, können in **Beschreibungssprachen** für **Vokabulare** definiert werden. Weit verbreitet ist das relativ einfache RDF Schema (**RDFS**), komplexere Sachverhalte können in der Web Ontology Language (**OWL**) ausgedrückt werden.

hbz

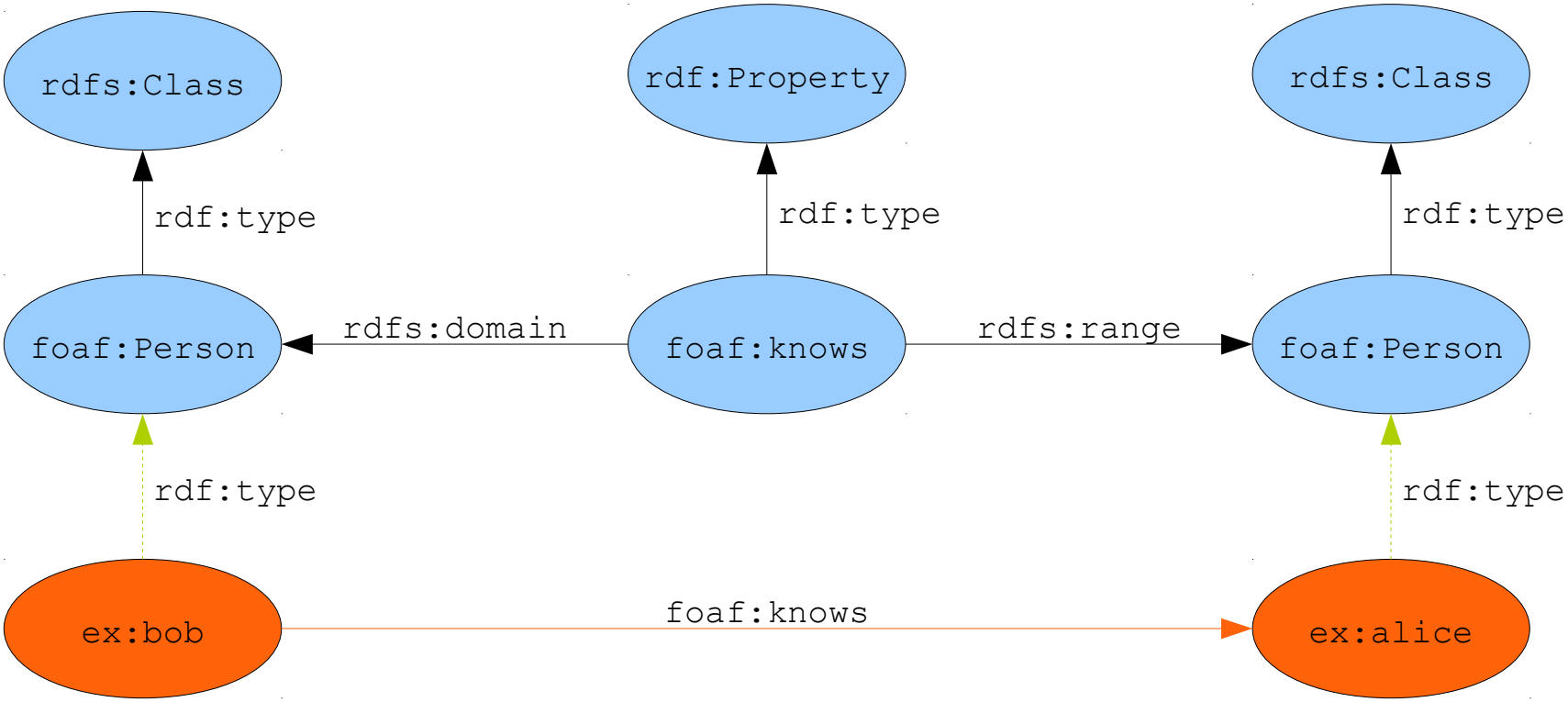
Auf Schema-Ebene werden **Klassen** definiert und es wird festgelegt, welche Beziehungen zwischen **Instanzen** dieser Klassen herrschen können. Mit dieser Information auf Vokabular-Ebene können **implizite Aussagen** auf Daten-Ebene **inferiert** werden.



(Wenn etwas ein Buch ist, dann ist es notwendiger Weise auch ein Artefakt; wenn etwas von jemandem geschrieben wurde, so wurde es auch von diesem erstellt. Wenn etwas von jemandem erstellt wurde, so ist dieses „etwas“ ein Artefakt, und dieser „jemand“ eine Person.)

hbz

RDF Schemata werden selbst in RDF ausgedrückt, und im Linked-Data-Paradigma erhalten die Klassen und Eigenschaften ebenfalls HTTP-URIs. So können auch sie mit einfachen Mitteln nachgeschlagen werden, wenn mehr Information zu ihnen benötigt wird.



(Das Prädikat rdf:type besagt, dass eine Resource eine Instanz einer Klasse ist. So sind ex:bob und ex:alice Personen, während Personen wiederum Klassen sind.)

hbz

```
# Explizite Tripel
ex:bob foaf:knows ex:alice .
```

```
# RDF Schema
foaf:knows   rdf:type     rdfs:Property ;
             rdfs:range   foaf:Person ;
             rdfs:domain foaf:Person .
foaf:Person rdf:type     rdfs:Class .
```

```
# Implizite Tripel, die aus dem Schema folgen
ex:bob   rdf:type foaf:Person .
ex:alice rdf:type foaf:Person .
```

RDF Schemata sind gut dafür geeignet, **Daten** aus verschiedenen Quellen zu **integrieren**, wenn sie unterschiedliche Vokabulare verwenden. Dazu werden die Klassen und Eigenschaften aus den verschiedenen Datenquellen zueinander in Beziehung gesetzt.

```
# Explizite Tripel
ex:bob ex:colleague ex:alice .
```

```
# RDF Schema als „Brücke“ zwischen Vokabularen
ex:colleague rdfs:subPropertyOf foaf:knows ;
             rdfs:domain        ex:Employee ;
             rdfs:range         ex:Employee .
ex:Employee  rdf:type           rdfs:Class ;
             rdfs:subClassOf    foaf:Person .
```

```
# Implizite Tripel, die aus dem Schema folgen
ex:bob   foaf:knows ex:alice .
ex:bob   rdf:type foaf:Person .
ex:alice rdf:type foaf:Person .
ex:bob   rdf:type foaf:Employee .
ex:alice rdf:type foaf:Employee .
```


hbz

Sie sind dran!

Erstellen Sie ein RDF Schema, so dass aus diesen Tripeln

```
@prefix team: <http://example.org/soccer/vocab#> .
@prefix   ex: <http://example.org/soccer/resource#> .

ex:team1 team:player ex:bob .
ex:team2 team:player ex:alice .
ex:game1 team:home ex:team1 .
ex:game1 team:away ex:team2 .
```

die folgenden Tripel geschlossen werden können

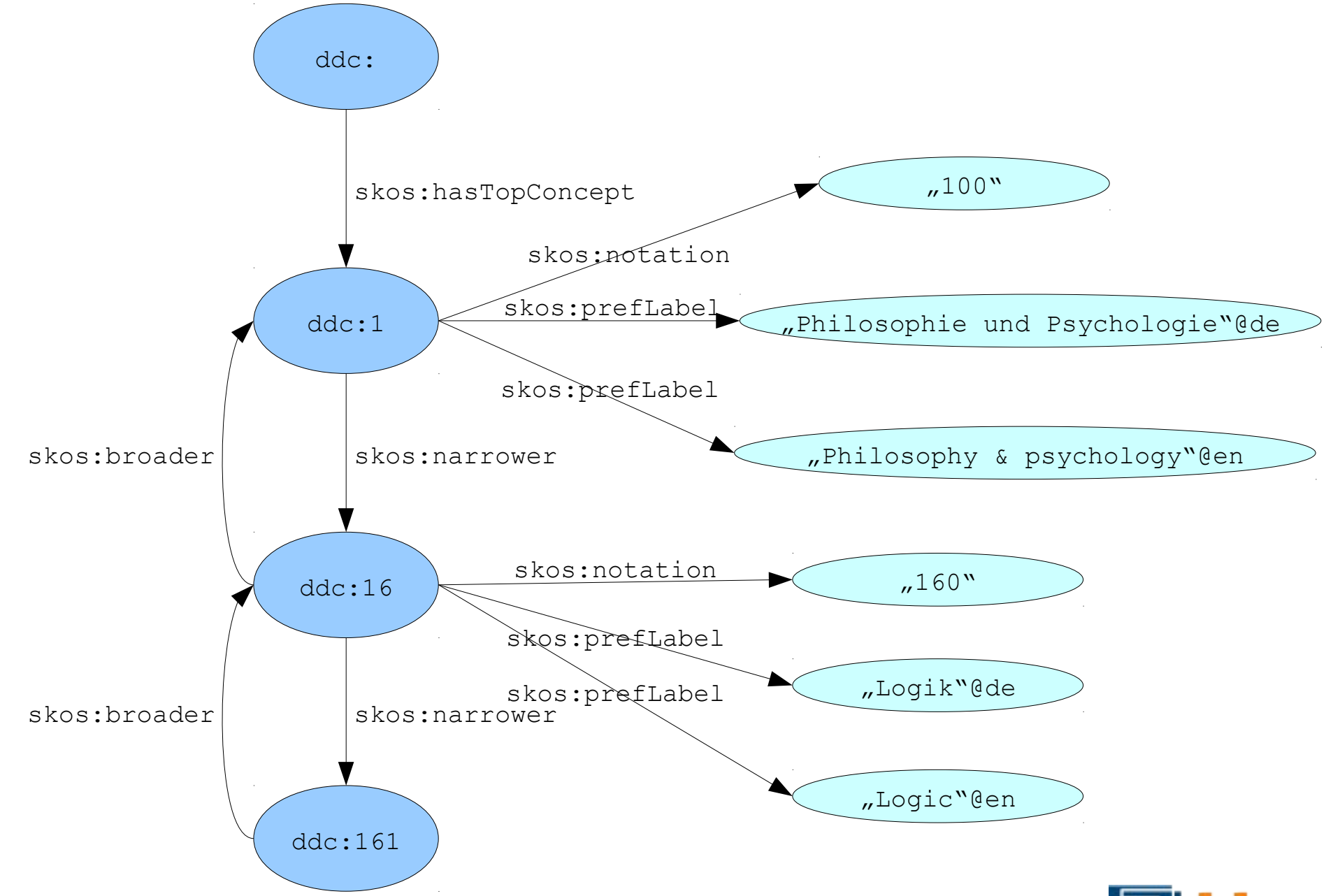
```
@prefix team: <http://example.org/soccer/vocab#> .
@prefix   ex: <http://example.org/soccer/resource#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .

ex:team1 rdf:type foaf:Group .
ex:team2 rdf:type foaf:Group .
ex:team1 foaf:member ex:bob .
ex:team2 foaf:member ex:alice .
ex:bob   rdf:type foaf:Person .
ex:alice rdf:type foaf:Person .
ex:game1 rdf:type team:Game .
ex:game2 rdf:type team:Game .
```


hbz

```
@prefix rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix team: <http://example.org/soccer/vocab#> .

team:player rdf:type rdfs:Property ;
            rdfs:subPropertyOf foaf:member ;
            rdfs:domain foaf:Person ;
            rdfs:range foaf:Group .

team:home   rdf:type rdfs:Property ;
            rdfs:domain team:Game .

team:away   rdf:type rdfs:Property ;
            rdfs:domain team:Game .

team:Game   rdf:type rdfs:Class .
```

Nicht immer werden die Ausdruckstärke und die Inferenzmöglichkeiten von RDFS oder OWL benötigt. Für klassische kontrollierte Vokabulare stellt das **Simple Knowledge Organization System** (SKOS) eine einfache, ebenfalls RDF basierte Alternative dar. Die Dewey Decimal Classification und die Library of Congress Subject Headings haben so schon ihren Weg in die Linked-Data-Welt gefunden.

hbz


ddc:
skos:hasTopConcept
„100“
skos:notation
ddc:1
skos:prefLabel
„Philosophie und Psychologie“@de
skos:prefLabel
„Philosophy & psychology“@en
skos:broader
skos:narrower
ddc:16
skos:notation
„160“
skos:prefLabel
„Logik“@de
skos:prefLabel
„Logic“@en
skos:broader
skos:narrower
ddc:161

hbz

Linked Data Principles

1) Use URIs as names for things.
2) Use HTTP URIs so that people can look up those names.
3) When someone looks up a URI, provide useful information, using the standards (RDF*, SPARQL).
4) Include links to other URIs. So that they can discover more things.

hbz

Fragen? Anmerkungen?

Diskussion!