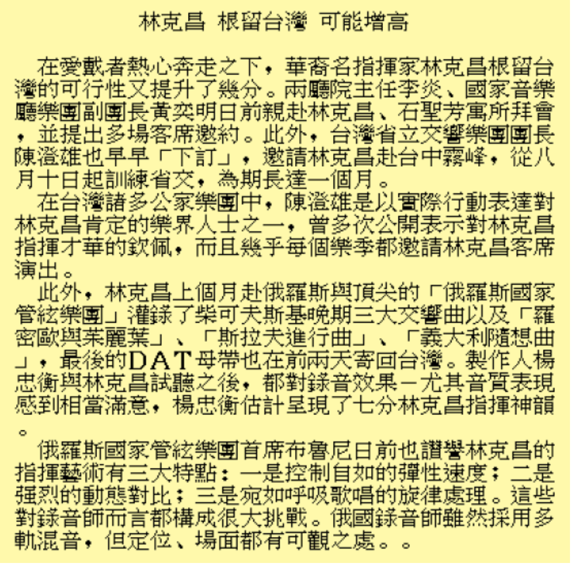
Knowledge Management with the WWW

Knowledge management must increasingly deal with Web-based knowledge, but the WWW does not support knowledge management well:

- **Main use by keyword-based search engines**
  - High recall, low precision
  - Results sensitive to vocabulary, not to intended meaning
  - Results are single Web pages
  - Location finding rather than information retrieval
  - Human agents required
- **Information of independent origin cannot be combined**
  - Coexistence of conflicting information
  - Coexistence of heterogeneous vocabularies
- **Implicit information cannot be derived**
  - No logical inferences
  - No metainformation about how to use (and not to use) Information

A Web Page Viewed by a Computer

How can you support meaningful processing without understanding the meaning of symbols like humans?



林克昌 根留台灣 可能增高

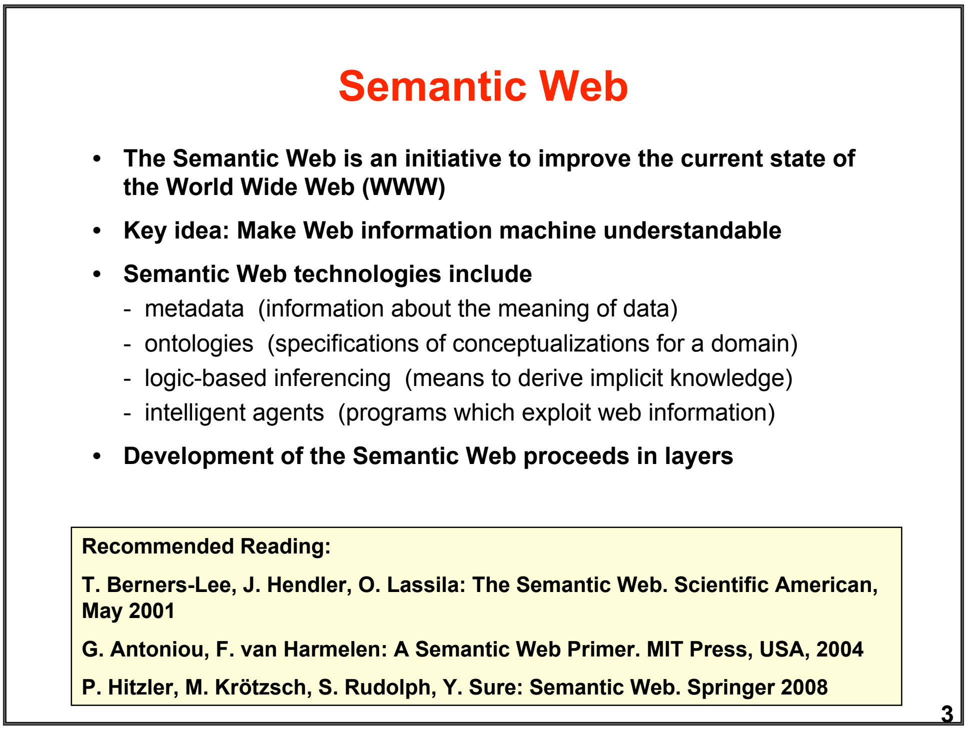
在愛戴者熱心奔走之下，華裔名指揮家林克昌根留台灣的可行性又提升了幾分。兩廳院主任李炎、國家音樂廳樂團副團長黃奕明日前親赴林克昌、石聖芳寓所拜會，並提出多場客席邀約。此外，台灣省立交響樂團團長陳澄雄也早早「下訂」，邀請林克昌赴台中霧峰，從八月十日起訓練省交，為期長達一個月。

在台灣諸多公家樂團中，陳澄雄是以實際行動表達對林克昌肯定的樂界人士之一，曾多次公開表示對林克昌指揮才華的欽佩，而且幾乎每個樂季都邀請林克昌客席演出。

此外，林克昌上個月赴俄羅斯與頂尖的「俄羅斯國家管絃樂團」灌錄了柴可夫斯基晚期三大交響曲以及「羅密歐與茱麗葉」、「斯拉夫進行曲」、「義大利隨想曲」，最後的ＤＡＴ母帶也在前兩天寄回台灣。製作人楊忠衡與林克昌試聽之後，都對錄音效果ㄧ尤其音質表現感到相當滿意，楊忠衡估計呈現了七分林克昌指揮神韻。

俄羅斯國家管絃樂團首席布魯尼日前也讚譽林克昌的指揮藝術有三大特點：一是控制自如的彈性速度；二是強烈的動態對比；三是宛如呼吸歌唱的旋律處理。這些對錄音師而言都構成很大挑戰。俄國錄音師雖然採用多軌混音，但定位、場面都有可觀之處。。

Slide originally presented by Frank van Harmelen



Semantic Web

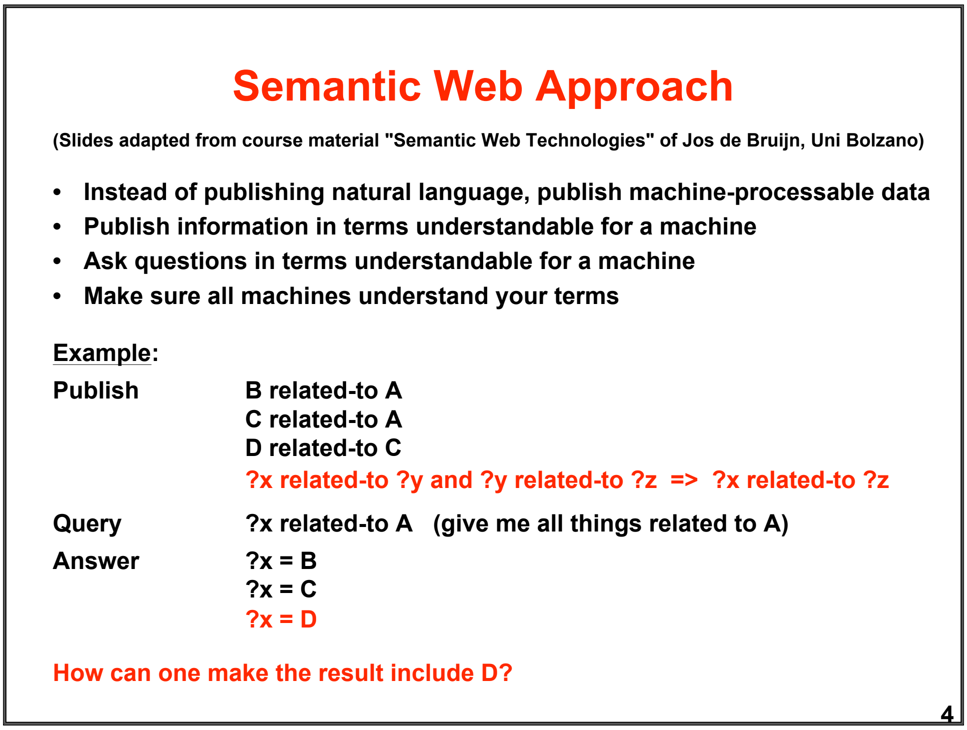
- The Semantic Web is an initiative to improve the current state of the World Wide Web (WWW)
- Key idea: Make Web information machine understandable
- Semantic Web technologies include
  - metadata (information about the meaning of data)
  - ontologies (specifications of conceptualizations for a domain)
  - logic-based inferencing (means to derive implicit knowledge)
  - intelligent agents (programs which exploit web information)
- Development of the Semantic Web proceeds in layers

Recommended Reading:

T. Berners-Lee, J. Hendler, O. Lassila: The Semantic Web. Scientific American, May 2001

G. Antoniou, F. van Harmelen: A Semantic Web Primer. MIT Press, USA, 2004

P. Hitzler, M. Krötzsch, S. Rudolph, Y. Sure: Semantic Web. Springer 2008



Semantic Web Approach

(Slides adapted from course material "Semantic Web Technologies" of Jos de Bruijn, Uni Bolzano)

- Instead of publishing natural language, publish machine-processable data
- Publish information in terms understandable for a machine
- Ask questions in terms understandable for a machine
- Make sure all machines understand your terms

Example:

Publish	B related-to A C related-to A D related-to C ?x related-to ?y and ?y related-to ?z => ?x related-to ?z
Query	?x related-to A (give me all things related to A)
Answer	?x = B ?x = C ?x = D

How can one make the result include D?

What are the Ingredients of the Web?

HTTP: **Hypertext Transfer Protocol**
(how to handle request-response pairs in the web)
e.g. GET /index.html

URI: **Universal Resource Identifier**
(how to address data)
e.g. http://kogs-www. informatik. uni-hamburg.de

HTML: **Hypertext Markup Language**
(how to mark up information to be read by humans)
e.g. <html> ... <head> ... <br>

What are the Ingredients of the Semantic Web?

XML (eXtensible Markup Language)
- **Provides a surface syntax for structured documents**
- **Does not provide semantic constraints on the meaning of documents**

XML Schema
- **Language for restricting the structure of XML documents**

RDF (Resource Description Framework)
- **Data model for objects ("resources") and relations between them**
- **Provides simple semantics for data model**
- **Can be expressed in XML**

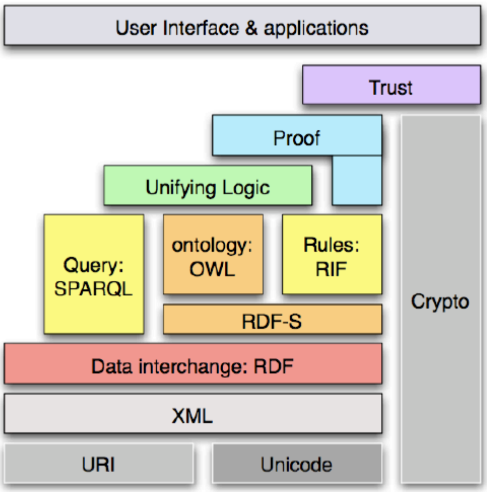
RDF Schema (RDFS)
- **Vocabulary description language for RDF resources**
- **Semantics for generalization hierarchies of properties and classes**

OWL (Web Ontology Language)
- **More expressive language for properties and relations between classes**

Semantic Web Layer Cake

(Term proposed by Berners-Lee)

Layers are developed bottom-up and standardized by "recommendations" of the World Wide Web Consortium (W3C)

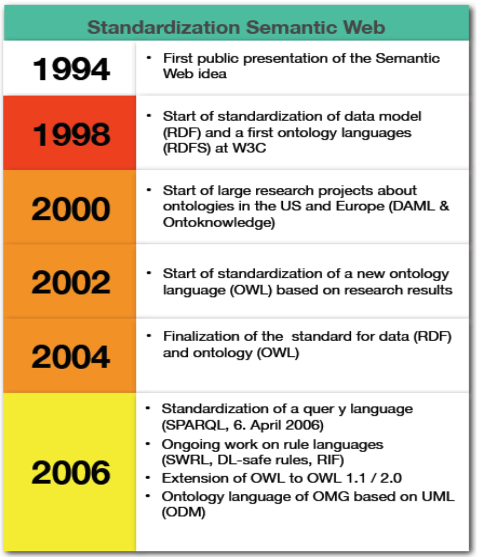


Standardization of the Semantic Web

Prerequisit for success of Semantic Web:

Open standards for defining the semantics of Web information

- clear definitions
- flexible
- extensible



Standardization Semantic Web	
1994	• First public presentation of the Semantic Web idea
1998	• Start of standardization of data model (RDF) and a first ontology languages (RDFS) at W3C
2000	• Start of large research projects about ontologies in the US and Europe (DAML & Ontoknowledge)
2002	• Start of standardization of a new ontology language (OWL) based on research results
2004	• Finalization of the standard for data (RDF) and ontology (OWL)
2006	• Standardization of a quer y language (SPARQL, 6. April 2006) • Ongoing work on rule languages (SWRL, DL-safe rules, RIF) • Extension of OWL to OWL 1.1 / 2.0 • Ontology language of OMG based on UML (ODM)

What is a Resource?

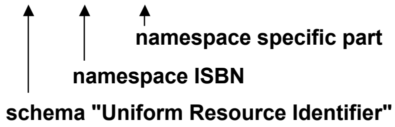
A resource may be:

- **A Web page** (e.g. http://www.w3.org/)
- **A person** (e.g. http://www.debruijn.net/)
- **A book** (e.g. urn:isbn:0-345-33971-1)
- **Anything denoted by a URI**

A URI is an <u>identifier</u> and not a location on the Web

RDF (Resource Definition Framework)

- **Infrastructure for using metadata between different web applications**
- **Allows making statements about resources (meta-data):**
- **Examples:** http://www.w3.org/ **has the format** text/html
  http://www.debruijn.net/ **has first name** Jos
  urn:isbn:0-345-33971-1 **has the author** Tolkien



namespace specific part

namespace ISBN

schema "Uniform Resource Identifier"

Structure of URIs

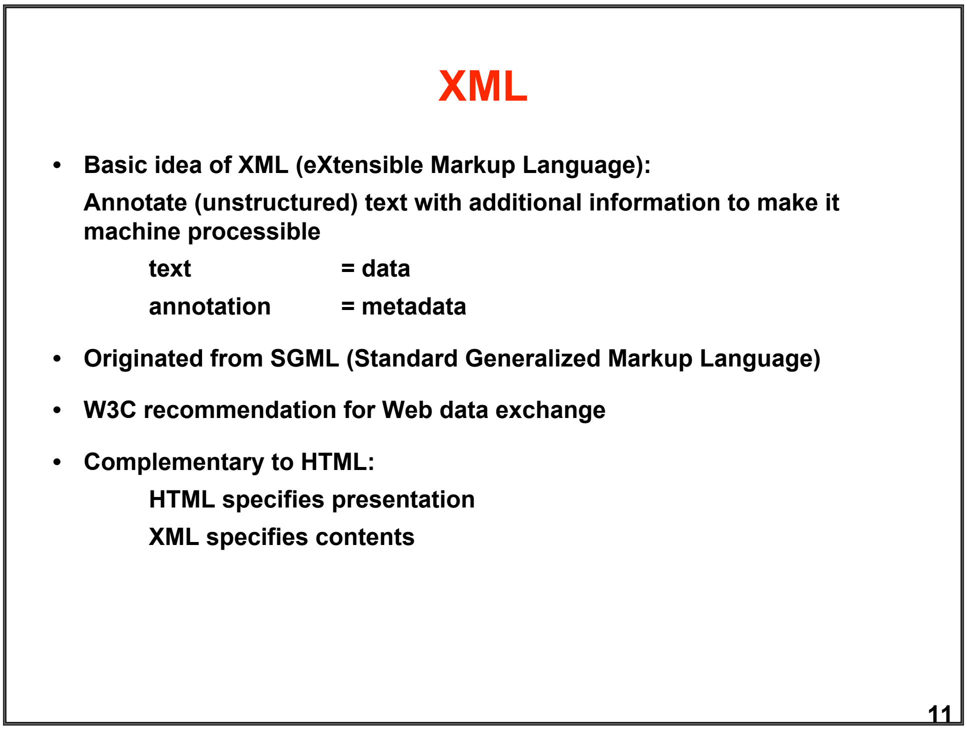
Structure of a URI:

Schema://[User [:Password]@]Server[:Port]/[Path][?Query][#Anchor]

Schema	**indicates type of URI, determines interpretation of the rest** **Examples:** ftp, http, mailto, urn
User, Password	**may be used for FTP authentication**
Server	**domain name or IP address of server**
Port	**optional: TCP port**
Path	**hierarchical identification of a resource within domain**
Query	**non-hierarchical identification within path target**
Anchor	**selection of fragment within document or data**

A URI with anchor is called a URI reference

URIs may be specified relative to a base URI, omitting
Schema://[User [:Password]@]Server[:Port]/ **and possibly parts of the path**



XML

- Basic idea of XML (eXtensible Markup Language):

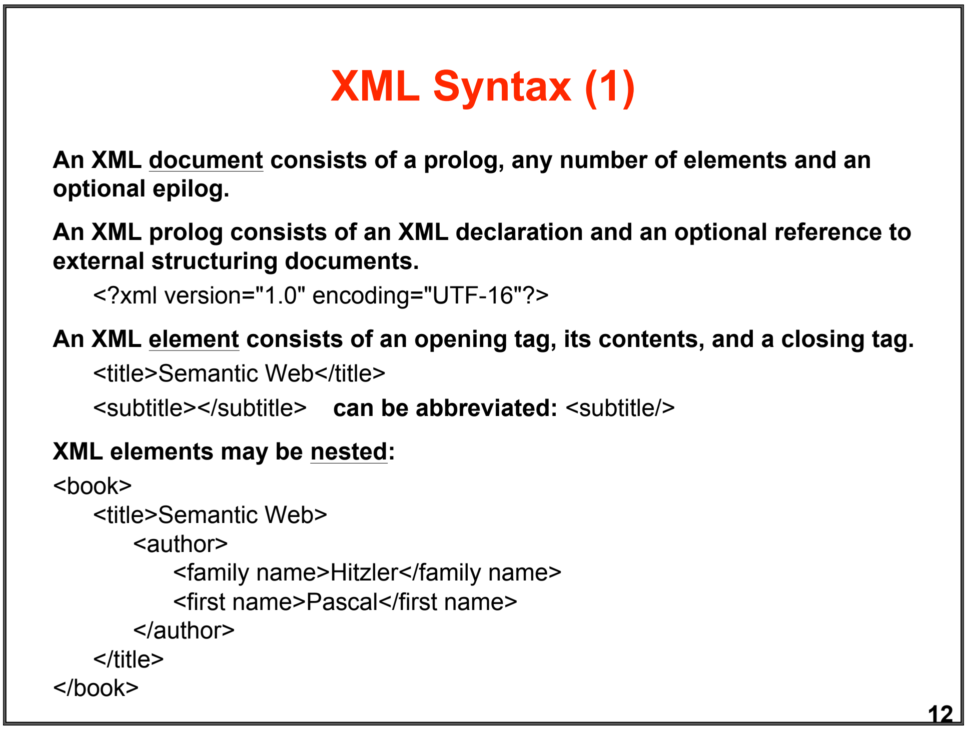
  Annotate (unstructured) text with additional information to make it machine processible

text	= data
annotation	= metadata

- Originated from SGML (Standard Generalized Markup Language)
- W3C recommendation for Web data exchange
- Complementary to HTML:

  HTML specifies presentation

  XML specifies contents



XML Syntax (1)

An XML document consists of a prolog, any number of elements and an optional epilog.

An XML prolog consists of an XML declaration and an optional reference to external structuring documents.

```
<?xml version="1.0" encoding="UTF-16"?>
```

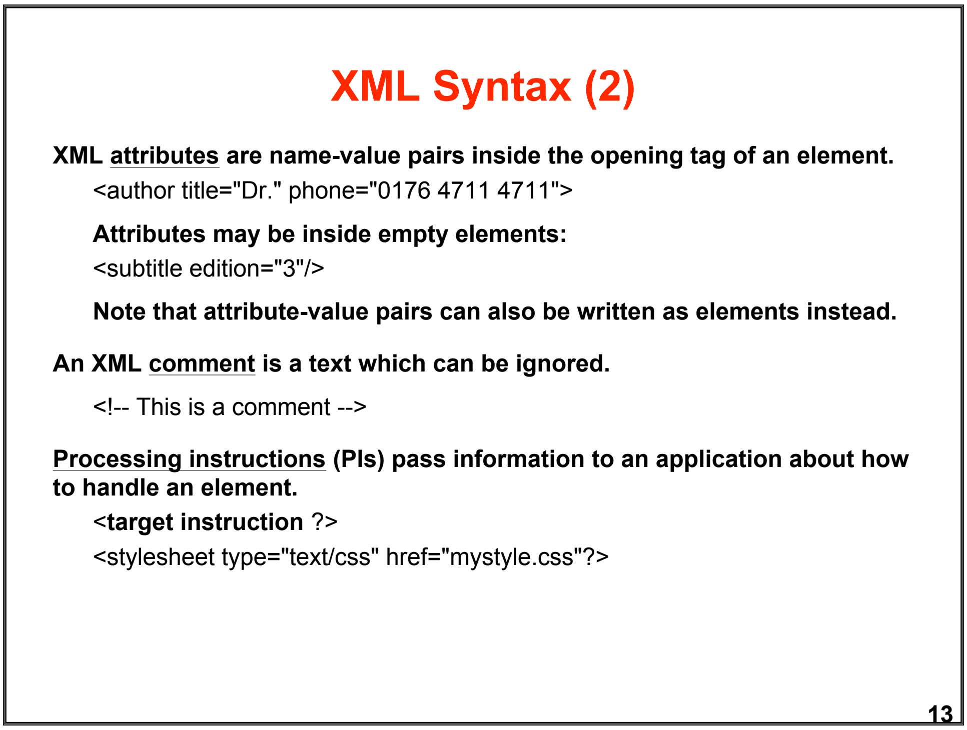
An XML element consists of an opening tag, its contents, and a closing tag.

```
<title>Semantic Web</title>
<subtitle></subtitle>
```

can be abbreviated: <subtitle/>

XML elements may be nested:

```
<book>
    <title>Semantic Web>
        <author>
            <family name>Hitzler</family name>
            <first name>Pascal</first name>
        </author>
    </title>
</book>
```



XML Syntax (2)

XML attributes are name-value pairs inside the opening tag of an element.

```
<author title="Dr." phone="0176 4711 4711">
```

Attributes may be inside empty elements:

```
<subtitle edition="3"/>
```

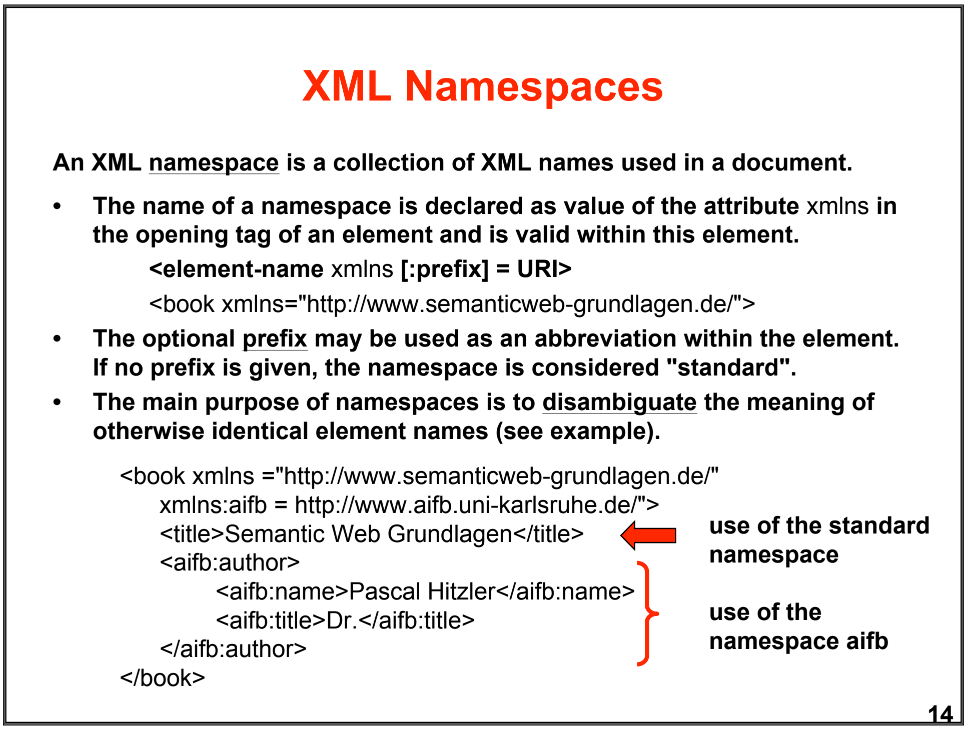
Note that attribute-value pairs can also be written as elements instead.

An XML comment is a text which can be ignored.

```
<!-- This is a comment -->
```

Processing instructions (PIs) pass information to an application about how to handle an element.

```
<target instruction ?>
<stylesheet type="text/css" href="mystyle.css"?>
```



XML Namespaces

An XML namespace is a collection of XML names used in a document.

- **The name of a namespace is declared as value of the attribute** xmlns **in the opening tag of an element and is valid within this element.**

  ```
  <element-name xmlns [:prefix] = URI>
  <book xmlns="http://www.semanticweb-grundlagen.de/">
  ```

- **The optional prefix may be used as an abbreviation within the element. If no prefix is given, the namespace is considered "standard".**
- **The main purpose of namespaces is to disambiguate the meaning of otherwise identical element names (see example).**

```
<book xmlns ="http://www.semanticweb-grundlagen.de/"
    xmlns:aifb = http://www.aifb.uni-karlsruhe.de/">
    <title>Semantic Web Grundlagen</title>
    <aifb:author>
        <aifb:name>Pascal Hitzler</aifb:name>
        <aifb:title>Dr.</aifb:title>
    </aifb:author>
</book>
```

use of the standard namespace

use of the namespace aifb

XML Schema

- **XML Schema is a language for defining the structure of XML documents beyond the structure imposed by basic XML.**
- **W3C recommendation complementing XML**
- **Downward compatible to DTDs (Document Type Definitions) inherited from SGML**
- **Syntax of XML Schema is defined in XML**
- **XML Schema definitions may be reused be extending or restring existing definitions.**

XML Schema Syntax

Opening tag of an XML Schema element (an "XML schema"):

```
<xsd:schema
    xmlns:xsd="http://www.w3.org/2000/10/XMLSchema"
    version="1.0">
```

Refers to the schema of XML Schema at the W3C Web site.

Element types:

<element name="n**ame**" **{optional attributes}**/>

<element name="head" minOccurs="1" maxOccurs="1"/>

Attribute types:

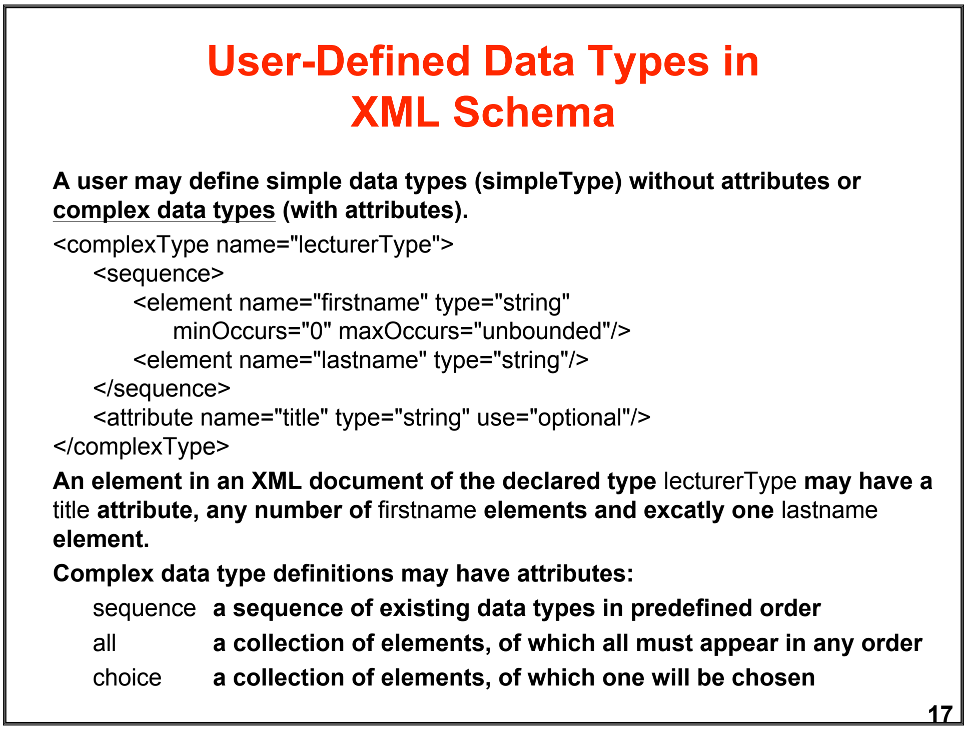
<attribute name="**name**" **{optional atrributes}**/>

<attribute name="id" type="ID" use="required"/>

Data types:

There are several built-in data types. e.g.

integer, Short, Byte, Long, Float, string, ID, IDREF, CDATA, Language



User-Defined Data Types in XML Schema

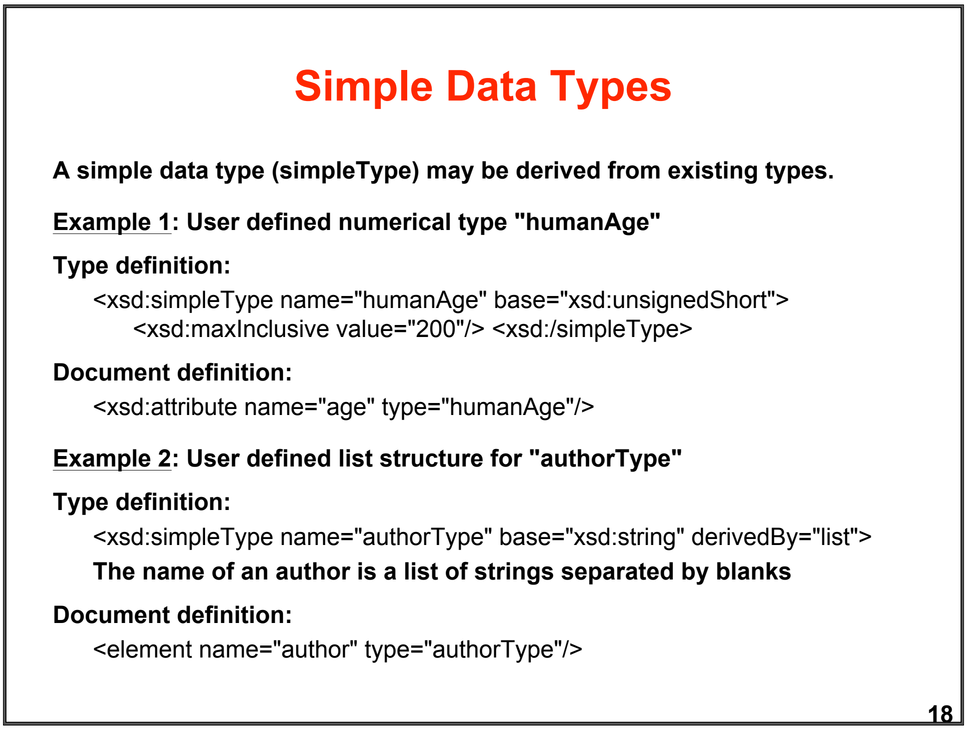
A user may define simple data types (simpleType) without attributes or <u>complex data types</u> (with attributes).

```
<complexType name="lecturerType">
    <sequence>
        <element name="firstname" type="string"
            minOccurs="0" maxOccurs="unbounded"/>
        <element name="lastname" type="string"/>
    </sequence>
    <attribute name="title" type="string" use="optional"/>
</complexType>
```

An element in an XML document of the declared type lecturerType **may have a** title **attribute, any number of** firstname **elements and excatly one** lastname **element.**

Complex data type definitions may have attributes:

sequence	**a sequence of existing data types in predefined order**
all	**a collection of elements, of which all must appear in any order**
choice	**a collection of elements, of which one will be chosen**



Simple Data Types

A simple data type (simpleType) may be derived from existing types.

<u>Example 1</u>: User defined numerical type "humanAge"

Type definition:

```
<xsd:simpleType name="humanAge" base="xsd:unsignedShort">
    <xsd:maxInclusive value="200"/> <xsd:/simpleType>
```

Document definition:

```
<xsd:attribute name="age" type="humanAge"/>
```

<u>Example 2</u>: User defined list structure for "authorType"

Type definition:

```
<xsd:simpleType name="authorType" base="xsd:string" derivedBy="list">
```

The name of an author is a list of strings separated by blanks

Document definition:

```
<element name="author" type="authorType"/>
```

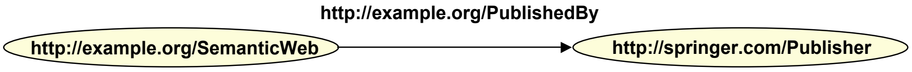
RDF

- RDF (Resource Description Framework) is a data model designed to enable a semantic data interpretation.
- Basic building blocks are object-attribute-value triples ("statements")
- RDF is domain-independent.
- Domain-specific terminology can be defined in RFS Schema (RDFS).
- Initially conceived as a metadata format, RDF is now widely accepted as a semantic data model.
- Basis for more expressive semantic languages (e.g. OWL)
- Used in diverse applications, e.g. for embedded information Adobe PDF-data or SVG vector graphics data.

Several slides adapted from course material "Semantic Web Technologies I" (WS 07/08) of M. Krötzsch, P. Hitzler, S. Rudolph

Building Blocks of RDF

Example of RDF-triple:


http://example.org/PublishedBy
http://example.org/SemanticWeb
http://springer.com/Publisher

Resources (identified by URIs)

- A URI identifies a resource, but does not necessarily point to it
- URIs correspond to nodes in a graph

  E.g. http://www.w3.org/1999/02/22-rdf-syntax-ns#Property

Properties (identified by URIs)

- Correspond to labels of edges in a graph
- Binary relations between two resources

  E.g. http://www.example.org/#hasName

Literals

- Concrete data values

  E.g. "John Smith", "1", "2006-03-07"

RDF Documents

To represent semantic net triples in RDF documents, resources are identified in one of three ways:

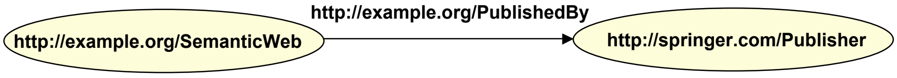
1. about attribute references an existing resource

```
<rdf:Description rdf:about="http://www.inf.uni-hh.de/~bn">
```

2. ID attribute creates a new resource

3. without a name creates an anonymous resource

RDF Descriptions (1)

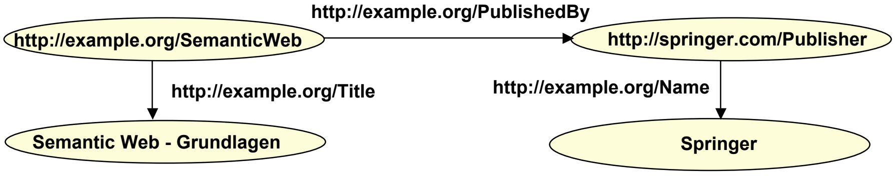

http://example.org/PublishedBy
http://example.org/SemanticWeb
http://springer.com/Publisher

Coding in RDF/XML:

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

- "Description" element encodes subject of a triple (URI of subject specified in "about" attribute)
- Each element nested in a description encodes a predicate (URI is specified by the name of the element)
- The object of nested predicates is specified by a description element within the predicate element

RDF Descriptions (2)

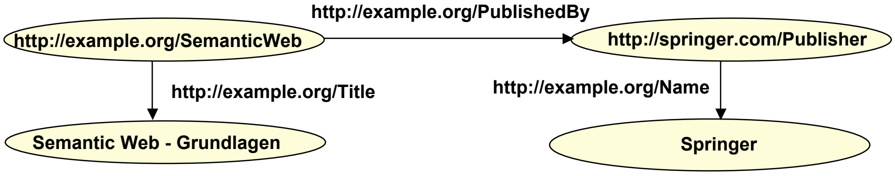

http://example.org/PublishedBy
http://example.org/SemanticWeb
http://springer.com/Publisher
http://example.org/Title
http://example.org/Name
Semantic Web - Grundlagen
Springer

Coding in RDF/XML:

```
<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</ex:Name>
        </rdf:Description>
    </ex:PublishedBy>
</rdf:Description>
```

- Untyped literals may be included in a predicate element as free text
- A subject element may contain several properties

RDF Descriptions (3)


http://example.org/PublishedBy
http://example.org/SemanticWeb
http://springer.com/Publisher
http://example.org/Title
http://example.org/Name
Semantic Web - Grundlagen
Springer

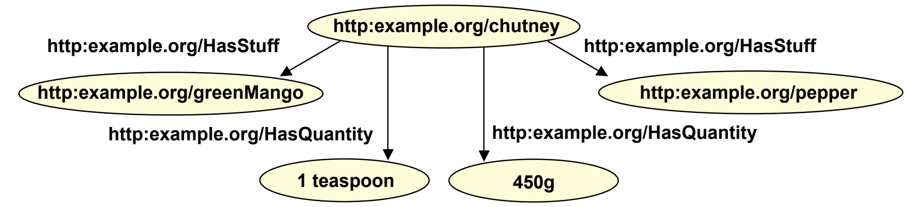
Alternative coding in RDF/XML:

```
<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" />
```

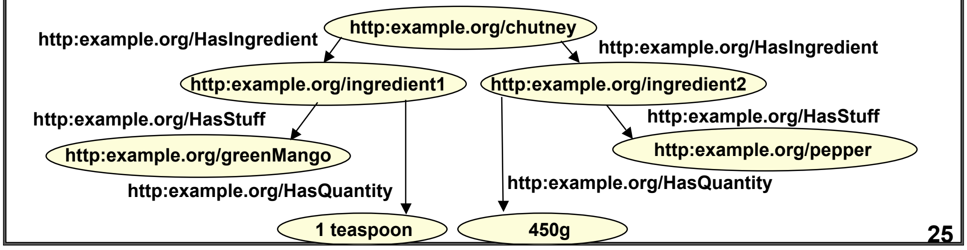
- Literals may be included in a predicate element as attributes.
- Attribute names specify property URIs.
- Object URIs are specified as the value of the rdf:resource attribute within a property tag.

Auxiliary and Blank Nodes (1)

"To prepare chutney one needs 450g green mango, 1 teaspoon pepper, ..."


http:example.org/chutney
http:example.org/HasStuff
http:example.org/HasStuff
http:example.org/greenMango
http:example.org/pepper
http:example.org/HasQuantity
http:example.org/HasQuantity
1 teaspoon
450g

This model is unsatisfactory: Quantities are not unambiguously associated with stuff! Reification of ingredients needed!


http:example.org/chutney
http:example.org/HasIngredient
http:example.org/HasIngredient
http:example.org/ingredient1
http:example.org/ingredient2
http:example.org/HasStuff
http:example.org/HasStuff
http:example.org/greenMango
http:example.org/pepper
http:example.org/HasQuantity
http:example.org/HasQuantity
1 teaspoon
450g

Auxiliary and Blank Nodes (2)

RDF/XML code:

```
<rdf:Description rdf:about="http://example.org/Chutney">
    <ex:hasIngredient rdf:nodeID="id1" />
    </rdf:Description>
    <rdf:Description rdf:nodeID="id1">
        <ex:ingredient rdf:resource="http://example.org/greenMango" />
        <ex:quantity>450g</ex:quantity>
    </ex:hasIngredient>
</rdf:Description>
```

Possible abbreviation:

```
<rdf:Description rdf:about="http://example.org/Chutney">
    <ex:hasIngredient rdf:parseType="Resource">
        <ex:ingredient rdf:resource="http://example.org/greenMango" />
        <ex:quantity>450g</ex:quantity>
    </ex:hasIngredient>
</rdf:Description>
```

RDF Schema (RDFS)

RDF allows object descriptions for resources (individuals)

➡ **Needed**:

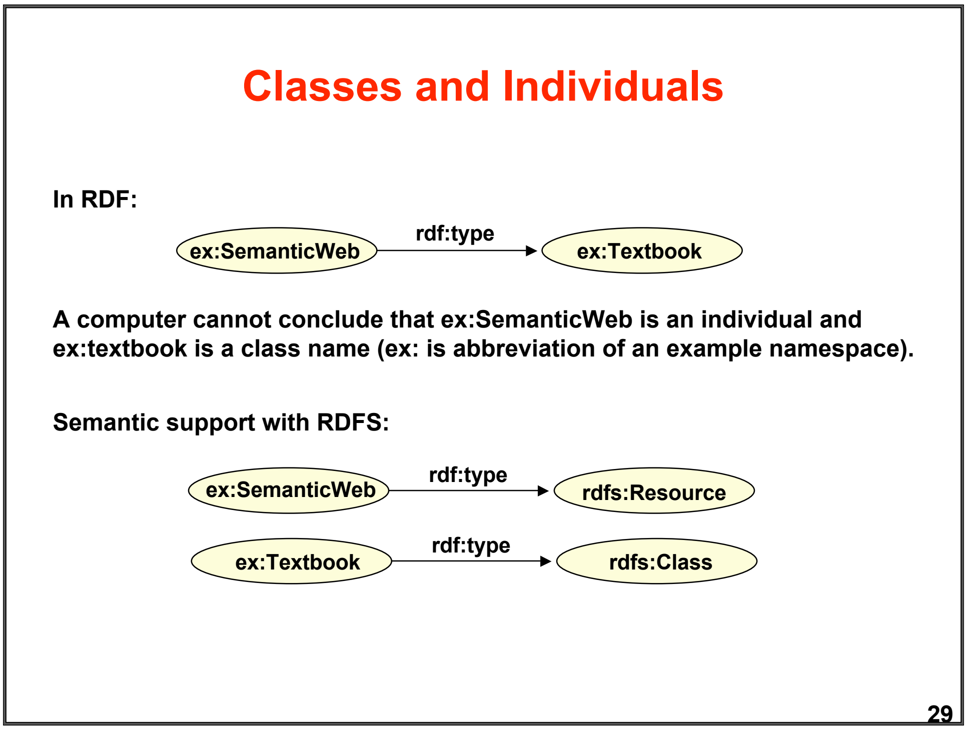
- Generic descriptions of object classes (concepts)
- Support for logical inferences

- RDFS is part of the W3C recommendation of RDF
- RDFS is a collection of RDF statements
- The vocabulary of RDFS is defined in the namespace
  http://www.w3.org/2000/01/rdf-schema#
  abbreviated with rdfs:

Semantics with RDFS

RDFS supports semantic processing:

- Domain-independent vocabulary
- Allows semantic annotation of domain-specific vocabulary ("metavocabulary")
- Software with RDFS support will correctly interpret any vocabulary defined by means of RDFS
- RDFS can be considered an ontology language for lightweight ontologies
- „A little semantics goes a long way.“



Classes and Individuals

In RDF:

ex:SemanticWeb —rdf:type→ ex:Textbook

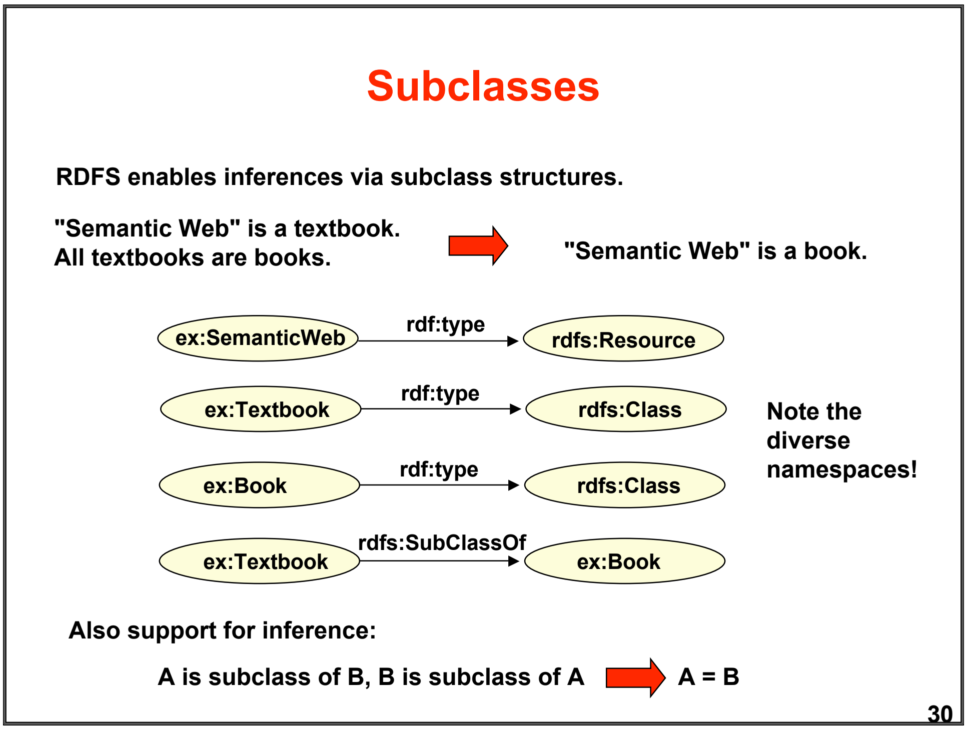
A computer cannot conclude that ex:SemanticWeb is an individual and ex:textbook is a class name (ex: is abbreviation of an example namespace).

Semantic support with RDFS:

ex:SemanticWeb —rdf:type→ rdfs:Resource

ex:Textbook —rdf:type→ rdfs:Class

29



Subclasses

RDFS enables inferences via subclass structures.

"Semantic Web" is a textbook.
All textbooks are books.

"Semantic Web" is a book.

ex:SemanticWeb —rdf:type→ rdfs:Resource

ex:Textbook —rdf:type→ rdfs:Class

ex:Book —rdf:type→ rdfs:Class

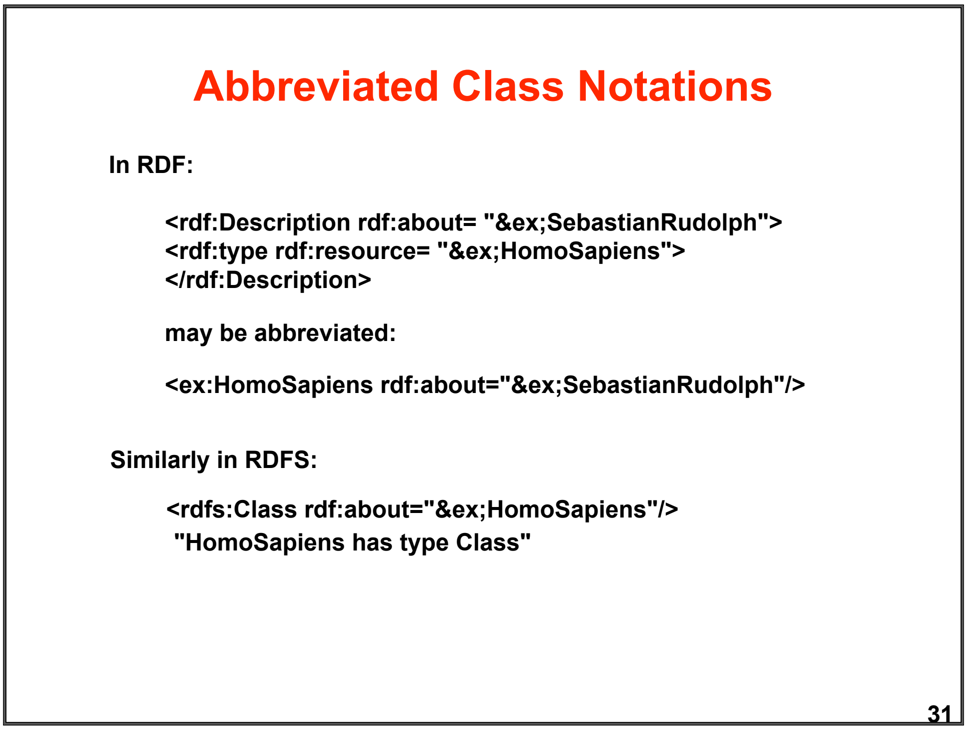
ex:Textbook —rdfs:SubClassOf→ ex:Book

Note the diverse namespaces!

Also support for inference:

A is subclass of B, B is subclass of A → A = B

30



Abbreviated Class Notations

In RDF:

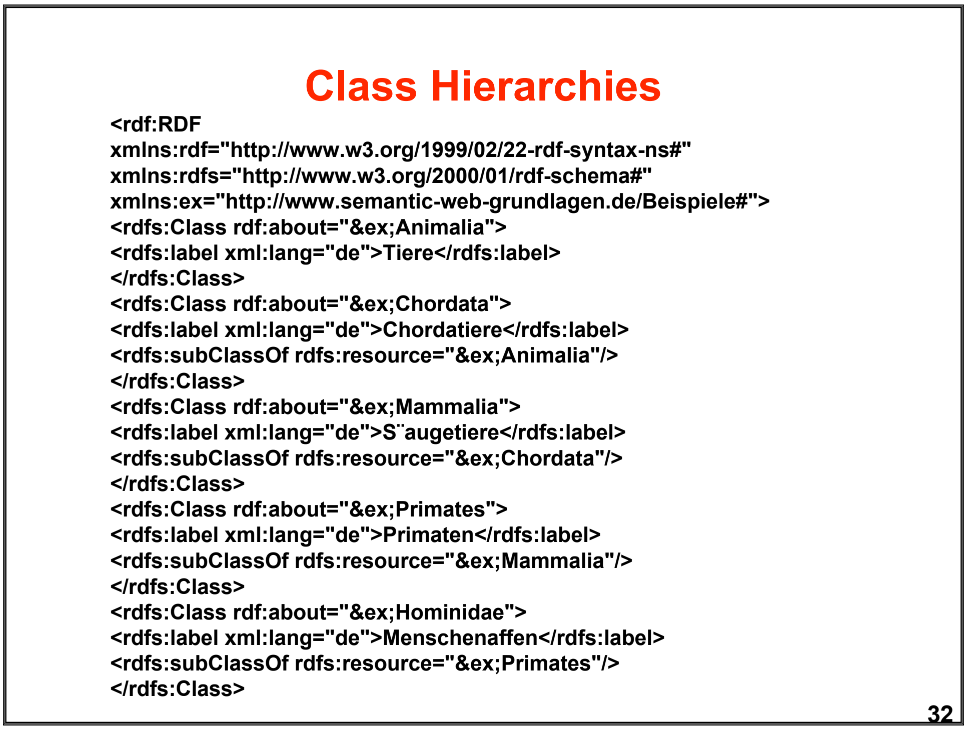
```
<rdf:Description rdf:about= "&ex;SebastianRudolph">
<rdf:type rdf:resource= "&ex;HomoSapiens">
</rdf:Description>
```

may be abbreviated:

```
<ex:HomoSapiens rdf:about="&ex;SebastianRudolph"/>
```

Similarly in RDFS:

```
<rdfs:Class rdf:about="&ex;HomoSapiens"/>
 "HomoSapiens has type Class"
```



Class Hierarchies

```
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:ex="http://www.semantic-web-grundlagen.de/Beispiele#">
<rdfs:Class rdf:about="&ex;Animalia">
<rdfs:label xml:lang="de">Tiere</rdfs:label>
</rdfs:Class>
<rdfs:Class rdf:about="&ex;Chordata">
<rdfs:label xml:lang="de">Chordatiere</rdfs:label>
<rdfs:subClassOf rdfs:resource="&ex;Animalia"/>
</rdfs:Class>
<rdfs:Class rdf:about="&ex;Mammalia">
<rdfs:label xml:lang="de">S¨augetiere</rdfs:label>
<rdfs:subClassOf rdfs:resource="&ex;Chordata"/>
</rdfs:Class>
<rdfs:Class rdf:about="&ex;Primates">
<rdfs:label xml:lang="de">Primaten</rdfs:label>
<rdfs:subClassOf rdfs:resource="&ex;Mammalia"/>
</rdfs:Class>
<rdfs:Class rdf:about="&ex;Hominidae">
<rdfs:label xml:lang="de">Menschenaffen</rdfs:label>
<rdfs:subClassOf rdfs:resource="&ex;Primates"/>
</rdfs:Class>
```

Reification (1)

How can one represent
"The detective believes that the butler has murdered the gardener" ?

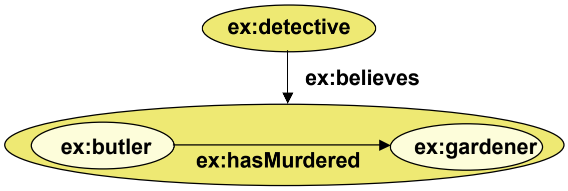
Unsatisfactory solutions:

ex:detective ex:believes "the butler has murdered the gardener"
ex:detective ex:believes ex:TheButlerHasMurderedTheGardener"

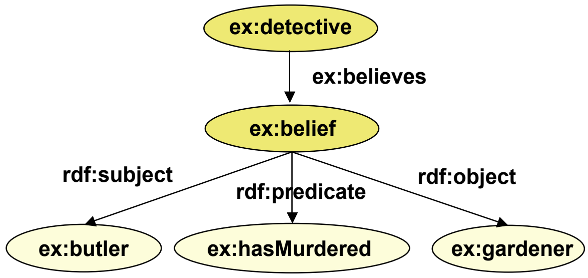
The object of the detective´s belief can be easily modelled:

ex:butler ex:HasMurdered ex:gardener

Solution: Reification of the detective´s belief

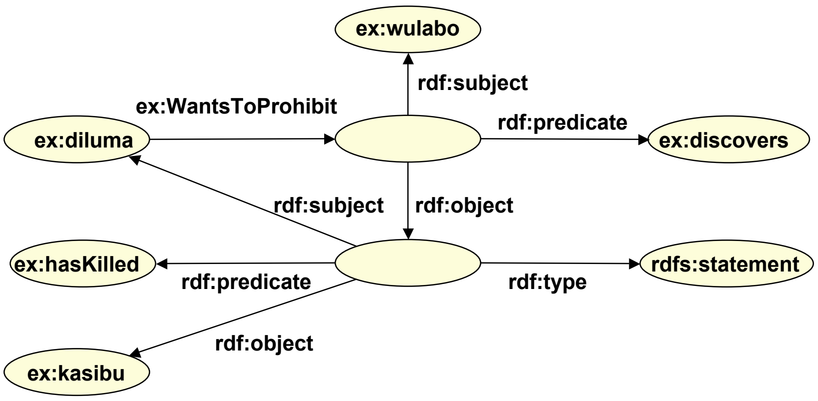


Reification (2)



Example with Reifications

What is the meaning of this diagram?



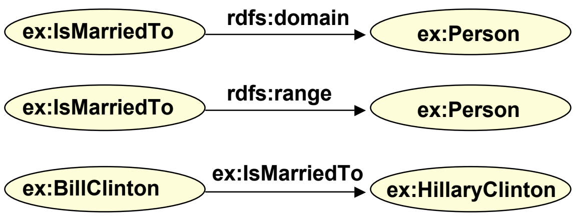
Domains and Ranges

RDFS enables inferences via domain and range information about properties.

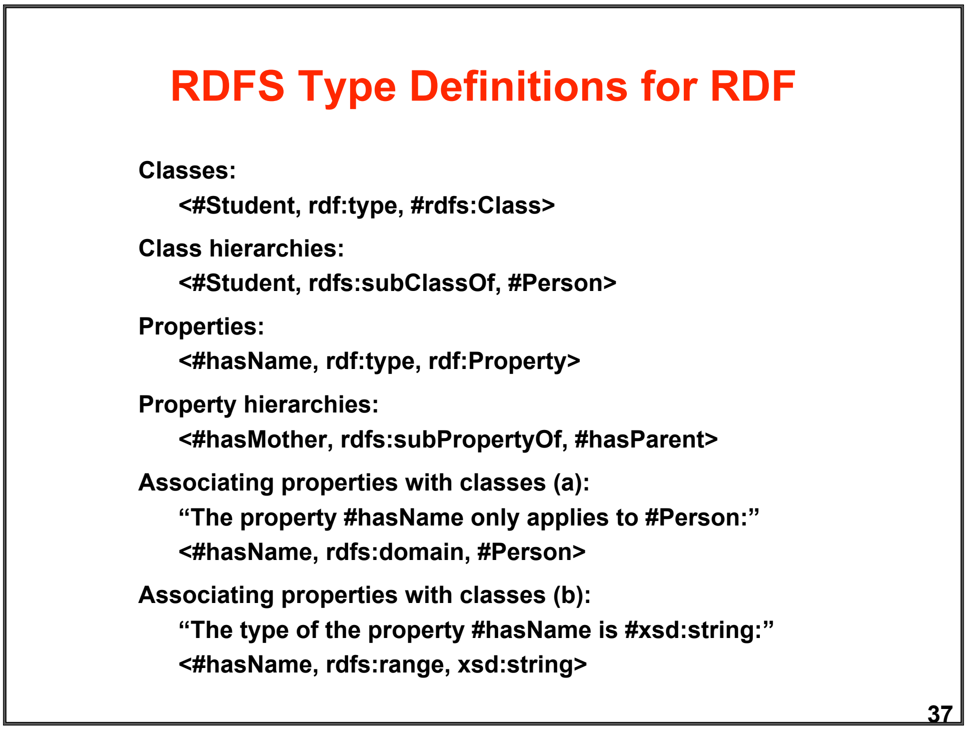
"Bill Clinton" is married to
"Hillary Clinton"

→

"Bill Clinton" is a person.
"Hllary Clinton" is a person.



Note that multiple domain or range definitions of the same property



RDFS Type Definitions for RDF

Classes:

<#Student, rdf:type, #rdfs:Class>

Class hierarchies:

<#Student, rdfs:subClassOf, #Person>

Properties:

<#hasName, rdf:type, rdf:Property>

Property hierarchies:

<#hasMother, rdfs:subPropertyOf, #hasParent>

Associating properties with classes (a):

“The property #hasName only applies to #Person:”

<#hasName, rdfs:domain, #Person>

Associating properties with classes (b):

“The type of the property #hasName is #xsd:string:”

<#hasName, rdfs:range, xsd:string>