Web 2.0
Lecture 10: Annotations

doc. Ing. Tomáš Vitvar, Ph.D.

tomas@vitvar.com • @TomasVitvar • http://vitvar.com

Czech Technical University in Prague

Faculty of Information Technologies • Software and Web Engineering • http://vitvar.com/courses/w20

Evropsky sociální fond
Praha & EU: Investujeme do vaší budoucnosti

Modified: Wed Mar 22 2017, 13:52:27
Humla v0.3

Data on the Web

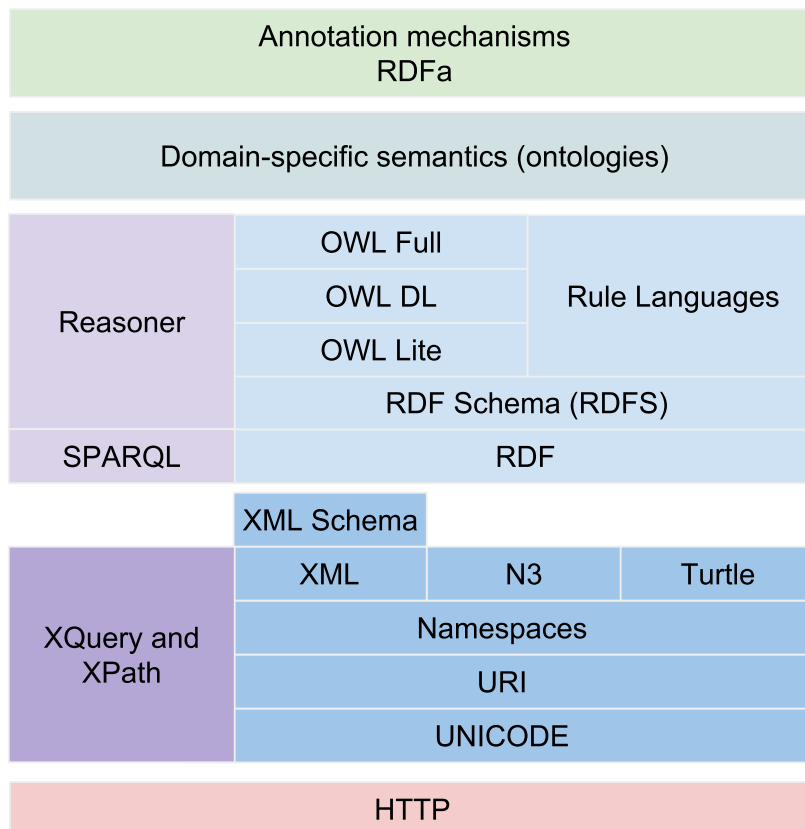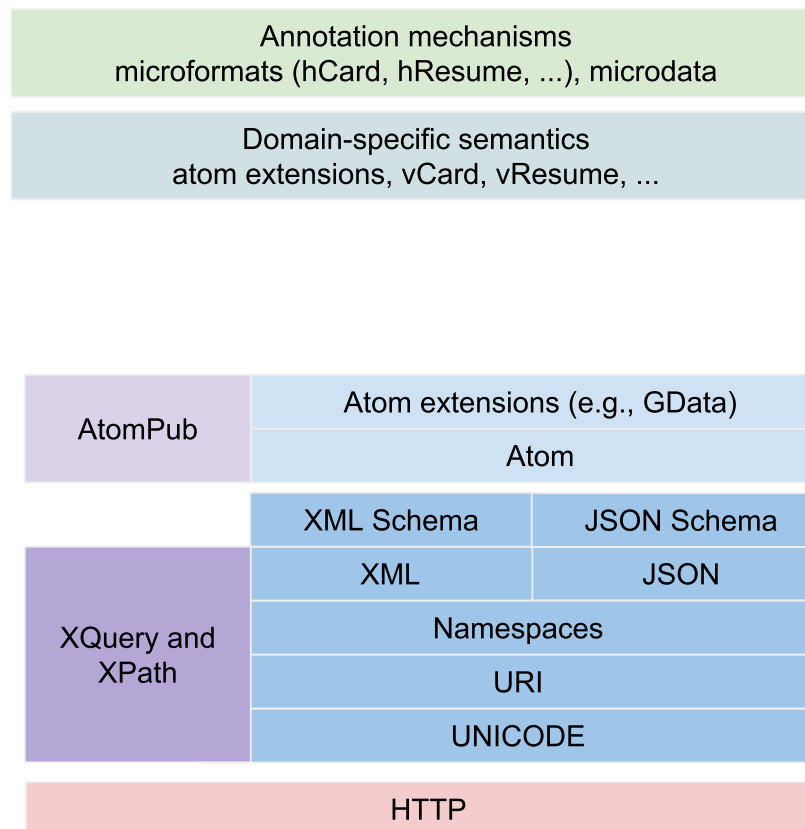| client	access	resource	back-end storage

browser, screen scraper — GET — *basic and digest access auth.* → **HTML**

crawler, third-party service — GET — *basic and digest access auth.* → **Annotated HTML (microformats)** [h*] [RDFa]

third-party service — GET, PUT, POST, DELETE — *AtomPub protocol* — *basic and digest access auth. OpenID 2.0 and OAuth 2.0* → **Atom Feed** [Atom]

third-party service — GET, PUT, POST, DELETE — *basic and digest access auth. OpenID 2.0 and OAuth 2.0* → **Proprietary formats** [XML] [JSON] [RDF]

API and Streaming API

Data Syntax, Structure and Semantics

Semantic Web Layered Cake
syntax and formal semantics

Annotation mechanisms RDFa

Domain-specific semantics (ontologies)

Reasoner	OWL Full	Rule Languages
	OWL DL	
	OWL Lite	
	RDF Schema (RDFS)	
SPARQL	RDF	

XQuery and XPath	XML Schema		
	XML	N3	Turtle
	Namespaces		
	URI		
	UNICODE		

HTTP

Web Data Formats
syntax and semantics (structure)

Annotation mechanisms microformats (hCard, hResume, ...), microdata

Domain-specific semantics atom extensions, vCard, vResume, ...

AtomPub	Atom extensions (e.g., GData)
	Atom

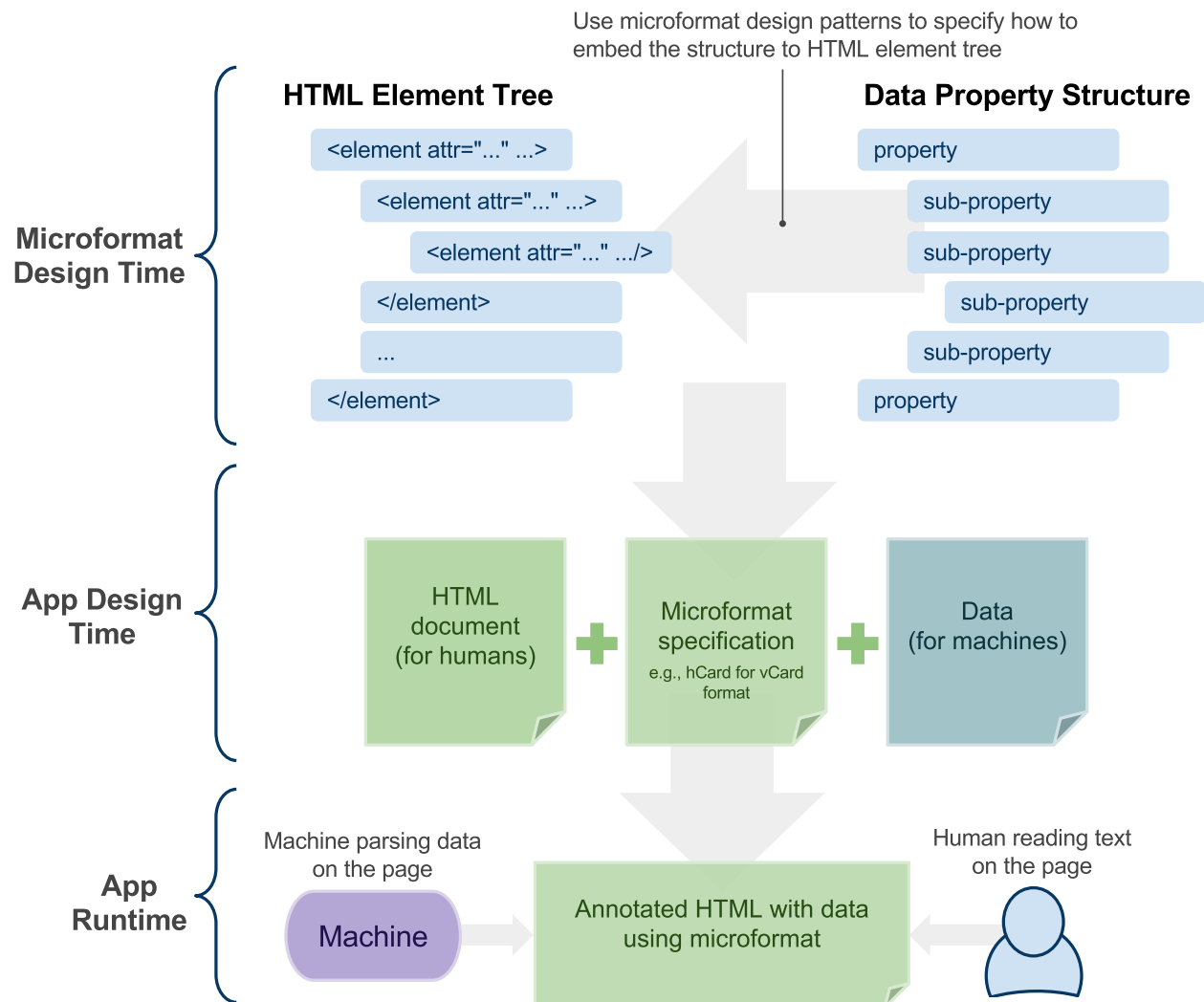| XQuery and XPath	XML Schema	JSON Schema
	XML	JSON
	Namespaces	
	URI	
	UNICODE	

HTTP

Overview

- Microformats
- Microdata
- RDF and RDFa
- OpenGraph Protocol

Microformats

- ## What is a microformat
  - *How to embed data in HTML, XHTML, Atom, and XML*
    - → *data: vCard, vResume, vRecipe*
    - → *micorformat: hCard, hResume, hRecipe*
  - *Browsers display HTML, machines process data*
  - *Microformat vs. POSH format*
    - → *POSH is same as microformat but data is not a standard format*

- ## Difference to Atom feeds
  - *Microformats require only a **single HTML document***
    - → *clients run GET to retrieve all data (human readable and machine readable)*
  - ***No significant increase of the size of document***
  - ***No requirements on data representation***
    - → *can be in any representation*
    - → *should be defined in a well-established format spec*

Microformats Usage

Use microformat design patterns to specify how to embed the structure to HTML element tree

HTML Element Tree

```
<element attr="..." ...>
    <element attr="..." ...>
        <element attr="..." .../>
    </element>
    ...
</element>
```

Data Property Structure

property
    sub-property
    sub-property
      sub-property
    sub-property
property

Microformat Design Time

App Design Time

HTML document (for humans)

+

Microformat specification
e.g., hCard for vCard format

+

Data (for machines)

App Runtime

Machine parsing data on the page

Machine

Annotated HTML with data using microformat

Human reading text on the page

Principles

- Design Patterns
  - *How to embed data in HTML elements or elements' attributes*
  - *Applied for a particular microformat specification*

- Follow semantics of (X)HTML elements
  - *Use the most appropriate semantic HTML element* ⬀
    - → *if not available, use* `<span>` *or* `<div>`

- XHTML Metadata Profiles (XMDP)
  - *Definition of metadata of a microformat in (X)HTML page*
  - *Machine and human readable, not a Web standard*
  - *Uses* `profile` *attribute on* `<head>` *element*
  - *Is deprecated in HTML5*
  - *Is an analogy to a namespace but not really a namespace!*
  - *See XHTML Metadata Profiles* ⬀ *specification*

vCard Example

- Describes contact information

```
1   PROPERTY:value1;value2;...;valueN
2   PROPERTY:SUBPROPERTY1="value";...SUBPROPERTY2="value";...
3
```

```
1    BEGIN:VCARD
2    VERSION:4.0
3    N:Vitvar;Tomas;Ing.;Doc.;PhD
4    FN: Doc. Ing. Tomas Vitvar, Ph.D.
5    ORG:Czech Technical University in Prague
6    TITLE:Associate Professor
7    PHOTO:http://vitvar.com/img/tomvit-portrait.jpg
8    TEL;TYPE="work,voice";VALUE=uri:tel:+420-2-334-334
9    TEL;TYPE="home,voice";VALUE=uri:tel:+420-2-443-554
10   ADR;TYPE=work;LABEL="Thákurova 6, Praha 6, Czech Republic"
11    :;;Thákurova 6;Praha 6;Czech Republic
12   EMAIL:tomas.vitvar@fit.cvut.cz
13   END:VCARD
14
```

- N – *a structured representation of the name (person/organization)*
- FN – *formatted name string*
- ORG – *name of the organization and associated units*
- TITLE – *job title, functional position*
- LABEL – *Addressing label*

Design Patterns Rules

- `class-design-pattern`
  - *semantic meaning indicated on HTML content by* `class` *attribute*

```
1   <div class="vcard">
2       <a class="url fn" href="http://www.vitvar.com">
3           Tomas Vitvar</a>,
4       <span class="org">UIBK</span>
5   </div>
```

- `value-class-pattern`
  - *embedding data structure when a property has subproperties*
    (vCard fragment is `TEL;TYPE=WORK:+43 554 554 556`*)*

```
1   <span class="tel">
2       <span class="type">Work</span>:
3       <span class="value">+43 554 554 556</span>
4   </span>
```

  - *sometimes value needs to be split into multiple pieces as follows*
    (note that dialing `+430554554556` *is not valid)*

```
1   <span class="tel">
2       <span class="type">Work</span>:
3       <span class="value">+43</span>(0)
4       <span class="value">554 554 556</span>
5   </span>
```

Design Patterns Rules (cont.)

- `include-pattern`
  - *to include a subset of data from one area of a page to the other area of the same page (same data to be reused by multiple microformats)*
  - ***cannot be used to include content from other URLs!***
  - *Example, a verbose hCard on a page:*

```
1  <div class="vcard" id="uibk-card">
2      <div class="fn org">University of Innsbruck</div>
3          <div class="adr">
4              <span class="street-address">Technikestrasse 21a</span>,
5              <span class="locality">Tirol</span>
6              <span class="postal-code">6020</span>
7          </div>
8      </div>
9  </div>
10
```

  - *Reviews on the same page:*
  (parser replaces the whole `<a>` *element including its content)*

```
1  <div class="hreview">
2      <h1 class="summary">A place to study computer science!</h1>
3      <div class="item"><a class="include" href="#uibk-card">Innsbruck Uni</a></d
4      <p class="description">Courses in English, fields of computer science.</p>
5  </div>
```

hCard Microformat Example

- hCard profile, options:

```
1   <link rel="profile" href="http://microformats.org/profile/hcard">
2
```

```
1   <...>This content uses <a rel="profile"
2       href="http://microformats.org/profile/hcard">hCard.</...>
3
```

```
1   <head profile="http://microformats.org/profile/hcard">..</head>
```

- Example specific rules
  - *vCard properties that do not make sense for hCard*
    - → *e.g., NAME, PROFILE, SOURCE, PRODID, VERSION*
    - → *publishers should not use them, parses should ignore them*
  - *if* `fn == org` *(i.e, `class="fn org"`)*
    - → *hCard is a contact for a company, organization or a place*
    - → `N` *(person's name) property should not be used or be the empty string*
  - *if* `fn != org` *AND* `fn` *contains two words*
    - → `fn` *is split into* `given-name` *and* `last-name`
    - → *sub-properties of* `N` *peoperty (by a whitespace or a comma)*
  - *see a complete specification in hCard Microformat Specification* ⧉

Known Issues

- ## Name conflicts and scalability
  - *More microformats on a page may cause naming conflicts*
    - → *no namespace support,* ***microformats do not scale***
    - → *functionality of tools may break when data formats change*

- ## No formal semantics, no reasoning support
  - *How important is it?*
  - *Semantics defined in XMDP profiles*
    - → *no formal basis though machine processable*
    - → *lack of compatibility with RDF/RDFa*
    - → *See Microformats and RDF/RDFa compatibility* ⬀ *for details.*

Uptake and some statistics

- Two billion pages annotated with hCard

- Google Rich Snippets
  - *Content indexing with microformats, microdata, RDFa*
    - *→ see Google Rich Snippets* ⧉
    - *→ 94% of the rich snippets data uses microformats*

Pizza **Pizzas Recipe** : Alton Brown : Food Network
www.foodnetwork.com › Recipes › Italian
★★★★★ 229 reviews - 24 hrs 45 mins
Food Network invites you to try this Pizza **Pizzas recipe** from Alton Brown.

- Firefox 3
  - *Native API to parse and process microformats in JavaScript*
  - *see Microformats support in Firefox 3* ⧉

- Facebook
  - *hCalendar and hCard for events*
  - *see Microformats in Facebook* ⧉

Overview

- Microformats
- Microdata
- RDF and RDFa
- OpenGraph Protocol

Microdata

- Part of HTML5 specification
  - *Google is the main driver (rich snippets support)*
  - *spec includes:*
    - → *Microdata vocabularies*
    - → *Microdata Global Attributes*
  - *see W3C working draft*

- Idea similar to microformats, but
  - *items (collection of properties) have ids (URIs)*
  - *Microdata vocabulary, a formal description of terms*
    - → `http://schema.org` *is becoming a standard*
    - → *e.g., Event, Organization, Person, Product, Review*
    - → *Created and supported by Google, Microsoft, Yahoo!*
    - → *have RDF representation too*
  - *data formats not directly based on formats such as vCard, vCalendar, they define its own "simple" vocabulary*

Global Attributes

- Attributes on any HTML element

- **Itemscope**
  - *identifies an element which descendants contain some properties*
    ```
    1 |  <div itemscope>...<div>
    ```

- **Itemtype**
  - *pointer to a vocabulary that describes the item and its properties*
    http://www.data-vocabulary.org/Person/

- **Itemid**
  - *global identifier of the item (URI)*
  - *such as a book's ISBN in urn schema,* urn:isbn:0-330-34032-8

- **Itemprop**
  - *a term from the vocabulary which value is in the element's content*
    ```
    1 |  <span itemprop="nickname">Johny<span>
    ```

- **Itemref**
  - *a reference to other item within the same document*
    ```
    1 |  <div itemscope itemref="myprofile"/>
    ```

Example

- Non-annotated HTML text

```
1  <section>
2    My name is Peter Brown and I work as a post-doc at the Innsbruck University.
3    My friends often call me Pete. My office address is
4    Technikestrasse 21a, 6020, Innbruck, and you can also visit my homepage at
5    <a href="http://peter-brown.org">http://peter-brown.org</a>
6  </section>
```

- Annotated HTML text with microdata

```
1  <section itemscope itemtype="http://schema.org/Person">
2    My name is <span itemprop="name">Peter Brown</span> and I work as a
3    <span itemprop="title">post-doc</span> at the
4    <span itemprop="affiliation">Innsbruck University</span>.
5    My friends often call me <span itemprop="nickname">Pete</span>.
6    <section itemprop="address" itemscope itemtype="http://schema.org/Address">
7      My office address is <span itemprop="street-address">Tchnikestrasse 21a</span>
8      <span itemprop="postal-code">6020<span>,
9      <span itemprop="locality">Innsbruck</span>
10   <section>
11   and you can also visit my homepage at
12   <a href="http://peter-brown.org" itemprop="url">http://peter-brown.org</a>
13 </section>
14
```

Microformats vs. Microdata

- ## Scalability
  - *Microformats specs are complicated because of specific rules tailored for vCard, vResume, etc.*
  - *Microdata can be easilly extensible, when new property occur they can be added witout breaking conformance of tools*

- ## Standards-based
  - *Microdata is a standard part of HTML5 effort*
  - *Microformats is an "ad-hoc" group of ethusiastic people, though widely supported*
    - *→ Strength is in underyling well-established formats*
  - *Microdata have links to Semantic Web efforts and Linked Data (via RDF), microformats not*

Overview

- Microformats

- Microdata

- RDF and RDFa
  - *Structured Property Values*
  - *Encoding RDF in XML (RDF/XML)*
  - *RDF–in attributes (RDFa)*

- OpenGraph Protocol

RDF

- Resource Description Framework (RDF)
  - *Resource – as defined in Web architecture*
    - → *usually anything that can be conveyed electronically*
    - → *plus abstract concepts that have no representation*
  - *RDF is at the bottom of Semantic Web stack of languages*

- References
  - *W3C Recommendations:*
    - → *RDF Suite of W3C Recommendations* ⬀,
    - → *RDF Primer* ⬀

Meaning of Data in XML

- A resource with URI `http://www.vitvar.com/data-about-me`

```
1   <person xmlns="http://example.org/people">
2       <name>Tomas Vitvar</name>
3       <mailbox>tomas@vitvar.com</mailbox>
4       <city>Innsbruck</city>
5   </person>
```

- No explicit meaning of terms
  - `person`, `name`, `mailbox`, … *are terms defined in namespace* `http://example.org/people` *but there is no URI assigned to them this does not work here:* `http://example.org/people#name`

- No explicit meaning of relationships
  - *a person <u>has</u> name <u>with value</u> Tomas Vitvar (→ Tomas Vitvar <u>is a</u> person), this person <u>has</u> mailbox <u>with value</u> tomas@vitvar.com (→ tomas@vitvar.com <u>is a</u> mailbox), etc.*
  BUT this person lives?, works?, was born?, … in a city Innsbruck

- Need for a language to describe statements
 → Resource Description Framework

RDF Statement

- RDF Tripple: **subject – predicate – object**
  - *a thing the statement describes (subject)*
  - *a specific property of the object (predicate)*
  - *a value of the property (subject)*

http://www.w3.org/2000/10/swap/pim/cont
act#fullname

http://vitvar.com#me

Tomas Vitvar
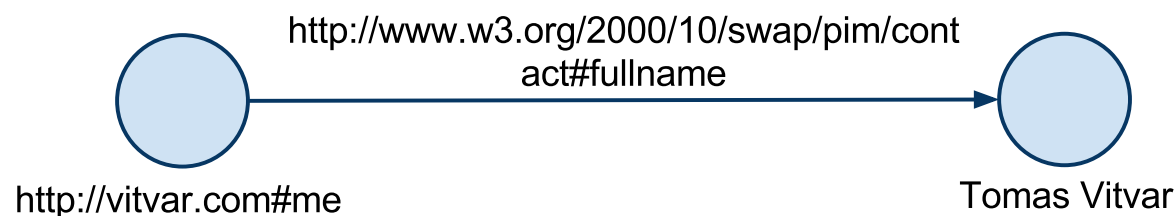
- Representation of statements
  - *using a graph notation*
    - → *nodes are subject and objects (rectangles are literals)*
    - → *arcs are predicates*
  - *identifiers to identify subject, predicate, object*
    - → *URI references (URIrefs)*
  - *machine processable language*
    - → *RDF serializations in triples, RDF/XML, N3, Turtle notations*

Meaning of Data in RDF
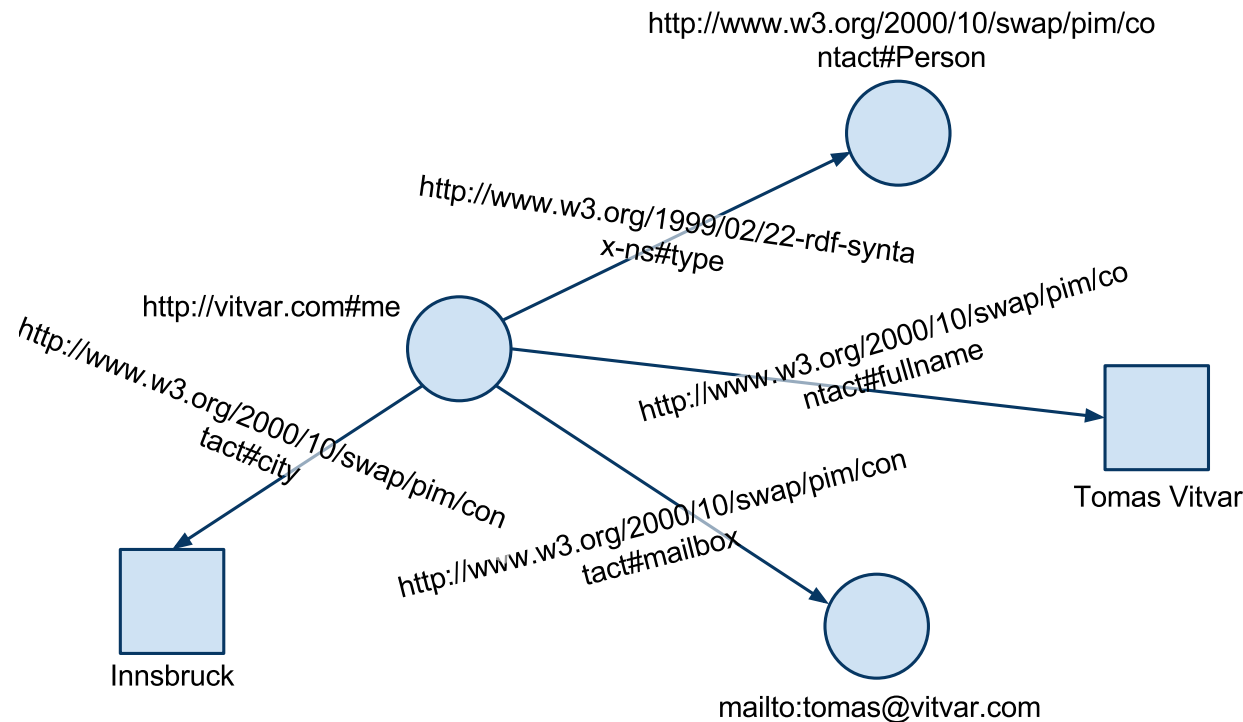
- **individuals:** Tomas Vitvar identified by `http://vitvar.com#me`
- **kinds of things:** Person identified by `#Person`
  - *properties* of those things, e.g., mailbox, identified by `#mailbox`
  - *values* of those properties, e.g. `mailto:tomas@vitvar.com`
    - *values of other data types such as strings, integers, dates, etc.*

References in statements

- URI identifies
  - *network-accessible things (electronic documents) → URL*
  - *things that are not network-accessible, such as human beings*
  - *abstract concepts that do not physically exist, such as "fullname"*
  - ***RDF uses <u>URI references</u> to identify subjects, predicates, objects***

- URI references (or URIref in short)
  - *URI with an optional fragment identifier*
  - `http://www.w3.org/2000/10/swap/pim/contact#fullname`
  - ***RDF resource is anything that can be identified with URIref***
  - *a set of URIrefs is called a **RDF vocabulary***

- Literals
  - *character strings to represent property values*
  - *can only be assigned to objects in RDF*
    (in other words, objects can be either URIrefs or literals)
    → they cannot be assigned to subjects or properties
  - *two kinds: **plain literals** and **typed literals***

RDF Serializations – Triples Notation

- Triples notation
  - *list of all triples from RDF graph*
  - *the full triples notation requires that URI references be written out completely (in angled brackets)*
  - *very long documents, some URIrefs need to be repeated*

```
1   <http://www.example.org/index.html> <http://purl.org/dc/elements/1.1/creator> "Toma
```

- Simplicity for examples
  - *QNames without angle brackets*
  - *Common prefixes and namespaces:*

```
1   rdf:  http://www.w3.org/1999/02/22-rdf-syntax-ns#
2   rdfs: http://www.w3.org/2000/01/rdf-schema#
3   dc: http://purl.org/dc/elements/1.1/
4   ex: http://www.example.org/
5   ext: http://www.example.org/terms
6   xsd: http://www.w3.org/2001/XMLSchema#
```

  - *example*

```
1   ex:index.html dc:creator "Tomas" .
2   ex:index.html dc:language "en" .
```

Kinds of Things

- Property `rdf:type`
  - *defines a type of a resource*

    ```
    1 | ex:me rdf:type ext:Person .
    ```

  - *corresponds to "is a member of" relationship*
  - `ext:Person` *understood as a class*
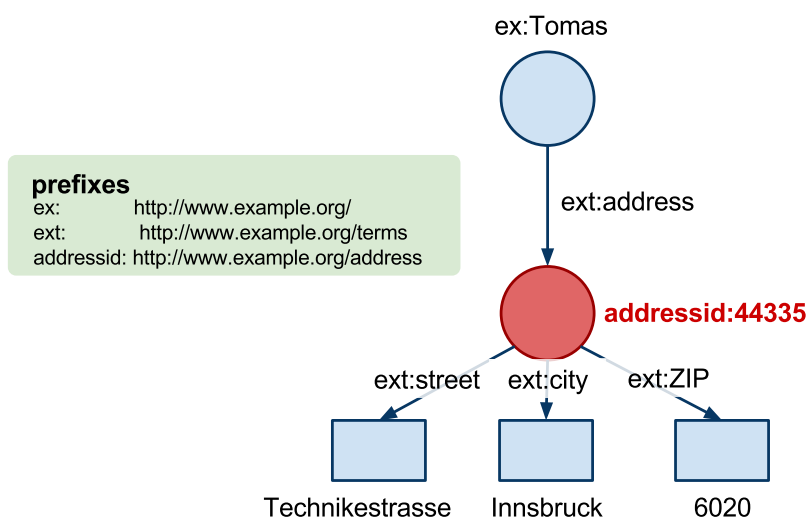   - → *however, RDF language does not define its semantics*
    - → *RDF Schema language provides additional vocabulary for class semantics*

Overview

- Microformats

- Microdata

- RDF and RDFa
  - *Structured Property Values*
  - *Encoding RDF in XML (RDF/XML)*
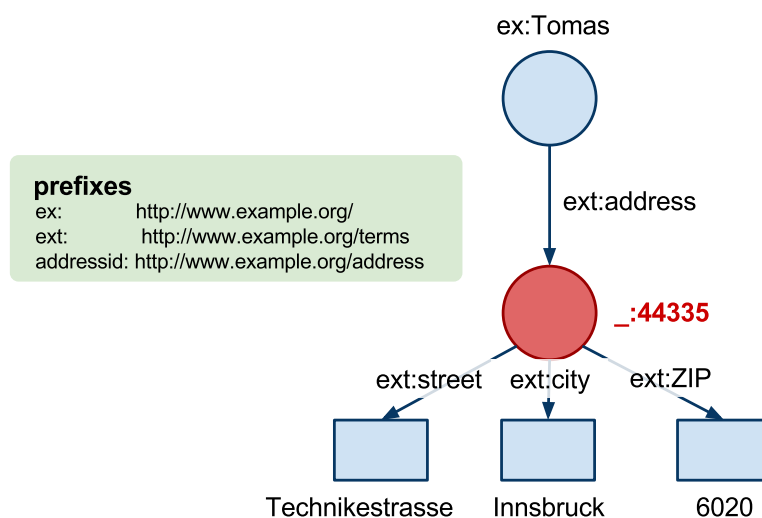 - *RDF–in attributes (RDFa)*

- OpenGraph Protocol

Structured Property Values

- Consider real-world complex structures
  - *Tomas works at Technikestrasse 21a, 6020 Innsbruck, Austria*
  - *One option to describe this using RDF:*

    ```
    1 | ex:Tomas ext:works "Technikestrasse 21a, 6020 Innsbruck, Austria" .
    ```

  - *But this is not often sufficient, such statements usually need to be recored as a structure, i.e. a street, a city, ZIP, ...*
    → *describe Tomas's* **address as a resource** *that has a URIref*

ex:Tomas

ext:address

prefixes
ex: http://www.example.org/
ext: http://www.example.org/terms
addressid: http://www.example.org/address

addressid:44335

ext:street   ext:city   ext:ZIP

Technikestrasse   Innsbruck   6020

Blank Nodes

- Does every structure need to have a URIref?
  - *When referenced from outside of the graphs yes, otherwise not*

- Blank nodes
  - *Nodes that do not need to be referenced from outside of the graph*
  - *No need for URIref, they are only used within the graph*

- Blank node identifier
  - *local within a graph: `_:LocalID`, must be uniqie within the graph*
  - *two blank nodes in two graphs with the same IDs are not the same!*

ex:Tomas

ext:address

prefixes
ex: http://www.example.org/
ext: http://www.example.org/terms
addressid: http://www.example.org/address

_:44335

ext:street   ext:city   ext:ZIP

Technikestrasse   Innsbruck   6020

Modeling with Blank Nodes

- N-ary relationships
  - *In fact, a blank node is a way to model an n-ary relationships*
  - *A blank node breaks down an n-ary to binary relationships*
  - *3-ary relationship between Tomas and (Technikestr, Innsbruck, 6020) Tomas – Technikestr, Tomas – Innsbruck, Tomas – 6020*

- Unidentified things
  - *not always good to use URIs such as e-mails to identify people*
    - → *e-mails may change, disappear, ...*
    - → *sometimes no need to assign unique ids to people*
  - *Example*
    - → *the author of the book is* `mailto:tomas@vitvar.com`*, as oposed to it is a person with e-mail* `mailto:tomas@vitvar.com`
  - *A person is an **abstract concept** that can be modeled using a blank node*

```
1   ex:book23 ext:author _:author1 .
2   _:author1 ext:email <mailto:tomas@vitvar.com .
3   _:author1 ext:name "Tomas Vitvar" .
4   _:author1 rdf:type ex:person .
```

Untyped and Typed Literals

- Untyped Literals
  - *No information about how to interpret a value of the plain literal*
  - *a programme must have a knowledge how to interpret the value*

```
1 | ex:person1 ext:age "24" .
```

- Typed literals
  - *pairing a string with a URIref that identifies a particular datatype*
    (`xsd:` refers to `http://www.w3.org/2001/XMLSchema#`)

```
1 | ex:person1 ext:age "24"^^xsd:integer
```

  - *RDF does not define its own data types (except `rdf:XMLLiteral`)*
    → *no need to map external to native ones*
  - *RDF uses external data types defined in XML Schema*
    → *not all are suitable, only basic ones such as `string`, `integer`, `date`*

Overview

- Microformats

- Microdata

- RDF and RDFa
  - *Structured Property Values*
  - *Encoding RDF in XML (RDF/XML)*
  - *RDF–in attributes (RDFa)*

- OpenGraph Protocol

Basic Rules

- Representation of RDF in XML language

- Example RDF triple
  - *a page* `index.html` *was created on August 16, 1999*

```
1 | ex:index.html   ext:creation-date   "Aug 16, 1999" .
```

- RDF/XML representation
  - *We can interpret a RDF statement as:*
    *a **description** that is **about** a subject of the statement*
  - *XML element (QName) of the description is the **predicate***
  - *a value of the element is the **object***

```
1  <?xml version="1.0"?>
2  <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
3      xmlns:ext="http://www.example.org/terms/">
4
5    <rdf:Description rdf:about="http://www.example.org/index.html">
6       <ext:creation-date>August 16, 1999</ext:creation-date>
7    </rdf:Description>
8  </rdf:RDF>
```

  - ***URIrefs must be written out when in attribute values***

Multiple Statements and Typed Literals

- Example RDF triples

```
1  ex:index.html   ext:creation-date   "Aug 16, 1999" .
2  ex:index.html   dc:language   "en" .
3  ex:index.html   ext:rank   "3"^^xsd:decimal .
4  ex:index.html   dc:creator   <http://www.vitvar.com#me> .
```

- RDF/XML representation

```
1  <?xml version="1.0"?>
2  <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
3      xmlns:dc="http://purl.org/dc/elements/1.1/"
4      xmlns:ext="http://www.example.org/terms/">
5
6      <rdf:Description rdf:about="http://www.example.org/index.html">
7          <ext:creation-date>August 16, 1999</ext:creation-date>
8          <dc:language>en</dc:language>
9          <ext:rank
10            rdf:datatype="http://www.w3.org/2001/XMLSchema#decimal">
11              3</ext:rank>
12         <dc:creator rdf:resource="http://www.vitvar.com#me"/>
13     </rdf:Description>
14  </rdf:RDF>
15
```

- *a description may combine all properties for a single subject but there also can be a description for every subject (such representations are the same)*

Blank Nodes

- Example RDF triples

```
1  ex:index.html   ext:editor   _:editor332 .
2  _editor332   ext:name   "Tomas Vitvar" .
3  _editor332   ext:homepage   <http://www.vitvar.com> .
```

- RDF/XML representation

```
1  <?xml version="1.0"?>
2  <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
3      xmlns:ext="http://www.example.org/terms/">
4
5      <rdf:Description rdf:about="http://www.example.org/index.html">
6          <ext:editor rdf:nodeId="editor332"/>
7      </rdf:Description>
8
9      <rdf:Description rdf:nodeId="editor332">
10         <ext:name>Tomas Vitvar<ext:name>
11         <ext:homepage rdf:resource="http://www.vitvar.com"/>
12     </rdf:Description>
13
14  </rdf:RDF>
```

   - *A node with id* `editor332` *can be referenced from within the RDF graph, not outside of the RDF graph*

Overview

- Microformats

- Microdata

- RDF and RDFa
  - *Structured Property Values*
  - *Encoding RDF in XML (RDF/XML)*
  - *RDF–in attributes (RDFa)*

- OpenGraph Protocol

RDFa

- Embedding RDF data in XHTML
  - *XHTML only, is extensible, HTML not*
    → RDFa defines a number of extension attributes
  - *Parses may recognize RDFa annotations in HTML too*
  - *RDFa is generic to embed arbitrary RDF data*
    → however, only standard (commonly agreed)
      vocabularies make sense

- W3C Recommendations:
  - *RDFa Specification* ⬈
  - *RDFa Primer* ⬈

Property and Object Values as Resources

- Creating a property using `rel` attribute
  - *assume, following text is at* `http://blog.vitvar.com/?p=107`

```
1   Content on this page is licensed under
2   <a xmlns:ext="http://www.example.org/terms"
3      rel="ext:license"
4      href="http://creativecommons.org/license/by/3.0">
5      a Creative Commons License – attribution
6   </a>
```

  - *This corresponds to the RDF triple*

```
1   <http://blog.vitvar.com/?p=107> ext:license
2       <http://creativecommons.org/license/by/3.0> .
```

  → *When the subject is not explicitly stated, then the subject is the URL of the XHTML page being described*

Property and Object Values as Literals

- Creating a property using `property` attribute
  - *RDFa defines a `property` extension attribute*
  - *assume, following text is at `http://blog.vitvar.com/?p=107`*

```
1   <div xmlns:dc="http://purl.org/dc/elements/1.1/">
2     <h3 property="dc:creator">Tomas<h3>
3   </div>
```

  - *This corresponds to the RDF triple*

```
1   <http://blog.vitvar.com/?p=107> dc:creator "Tomas" .
```

- Typed literals
  - *RDFa defines a `datatype` extension attribute*

```
1   <div xmlns:dc="http://purl.org/dc/elements/1.1/"
2           xmlns:xsd="http://www.w3.org/2001/XMLSchema#">
3     <h3 property="dc:creator" datatype="xsd:string">Tomas<h3>
4   </div>
```

- Alternative content
  - *RDFa defines `content` extension attribute*
    - → *replaces the object value that is in the element's value*

```
1   <div xmlns:dc="http://purl.org/dc/elements/1.1/">
2     <h3 property="dc:date" content="2011-04-08">8 April<h3>
3   </div>
4
```

Subject

- Creating a subject using `about` attribute
  - *RDFa defines* `about` *extension attribute*
  - *Let the following text is at* `http://blog.vitvar.com/?p=107`

    ```
    1   <div xmlns:dc="http://purl.org/dc/elements/1.1/"
    2           about="/p/107" >
    3     <h3 property="dc:creator">Tomas<h3>
    4   </div>
    ```

  - *This corresponds to the RDF triple*

    ```
    1   <http://blog.vitvar.com/p/107> dc:creator "Tomas".
    ```

  - *Also possible to use multiple subjects on a single page*

Types and Blank Nodes

- Types
  - *RDFa defines* `typeof` *extension attribute*
    - → *corresponds to* `rdf:type` *property*

- Blank node
  - *When annotation has* `typeof` *but not* `about`
    - → *blank node, that is, a node without a subject*

```
1  <div about="#me" rel="foaf:knows"
2    xmlns:foaf="http://xmlns.com/foaf/0.1">
3  <div typeof="foaf:person">
4    <p property="foaf:name">Peter<p>
5    <p>
6      Email: <a rel="foaf:mbox" href="mailto:peter@novak.cz">
7        peter@novak.cz</a>
8    <p>
9  </div>
```
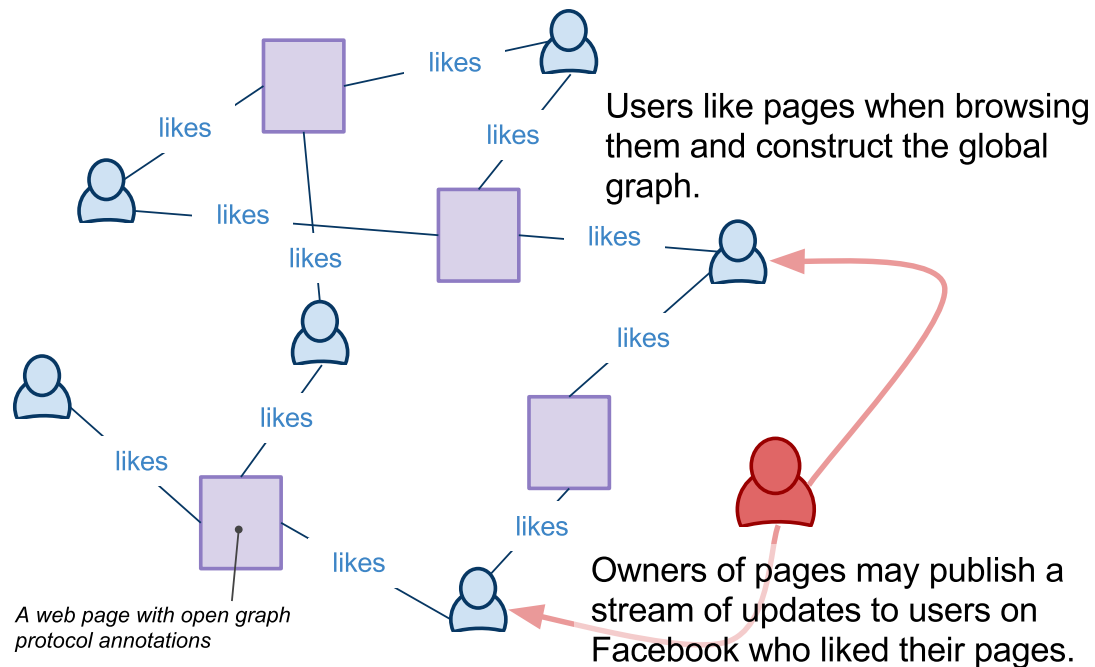
  - *I know Peter who has e-mail petr@novak.cz*

Overview

- Microformats
- Microdata
- RDF and RDFa
- OpenGraph Protocol

OpenGraph Protocol

- Global Social Graph
  - *important adoption of RDFa, see Open Graph Protocol* ⬈
  - *defines meta-data for pages' description so that it can be easily included in a global graph connecting people and pages through "likes" (a person – likes – a page)*

Users like pages when browsing them and construct the global graph.

Owners of pages may publish a stream of updates to users on Facebook who liked their pages.

A web page with open graph protocol annotations

Page Annotations

- Open Graph protocol main properties
  - *a page is the subject in the RDF triple*
  - `og:title` – *title of the page*
  - `og:type` – *type of the content (e.g., movie)*
  - `og:image` – *URL of the image for the page*
  - `og:url` – *a canonical URL of the page to be used as its permanent ID in the graph*

- HTML page annotation RDFa example

```html
1   <html xmlns:og="http://ogp.me/ns#">
2     <head>
3       <title>The Rock (1996)</title>
4       <meta property="og:title" content="The Rock" />
5       <meta property="og:type" content="movie" />
6       <meta property="og:url" content="http://www.imdb.com/title/402/" />
7       <meta property="og:image" content="http://media-imdb.com/rock.jpg" />
8       ...
9     </head>
10    ...
11  </html>
```

Publishing updates

- Ownership
  - *Page must be associated with a Facebook application*
    - → *using* `fb:app_id` *meta tag*
  - *Owners can publish a stream of updates using the Facebook Graph API*

- Getting access

```
1   curl -F type=client_cred \
2       -F client_id=app_id \
3       -F client_secret=app_secret \
4       https://graph.facebook.com/oauth/access_token
```

- Publishing updates

```
1   curl -F 'access_token=...' \
2       -F 'message=Hello Likers' \
3       -F 'id=http://www.mydomain.com/great_page.html' \
4       https://graph.facebook.com/feed
```