Introduction to Linked Data

Sandro Hawke, W3C
sandro@hawke.org
@sandhawke

http://www.w3.org/2010/Talks/0608-linked-data
June 8 2010, Cambridge Semantic Web Gathering
Outline

- Context
- Motivation
- Prerequisites (Web, RDF)
- Using Live URIs as Identifiers
- Related and Future Work
- Questions
Technology Context

Semantic Web Technologies
- RDF
- SPARQL
- SKOS
- OWL
- DL
- Logic Programming
- Theorem Proving
- Entity Extraction
- Coreference
- Speech Recognition
- KR

Linked Data
- RIF
- GRDDL
- XML
- XPath
- URIs
- HTTP
- SSL
- CSS
- HTML
- linkback
- AJAX
- svg
- png
- WebApps
- JavaScript
- WCAG

Web Technologies
- XSLT
The Flow of Data

- Web Data Apps
- Inference
- Linked Data
- RDF Triples
- Data in SQL DBs
- Data in Spreadsheets
- XML Data
- Raw Sensor Data
- Crowd source Data
- Etc...
So apps can find data

... without centralization:
  - No central bottlenecks
  - No central point-of-failure
  - No central policies (privacy? security)?
  - No need for permission

Use existing social structures

Use existing (Web) technology structures
Anyone Can Say Anything

- Whatever your business, projects, thoughts
- ... and others can find/filter/analyze/reuse
So How Do We Do It?

- **Be Linkable**  
  *(Good Website Design)*

- **Show Your Triples**  
  *(Export as RDF)*

- **Use Live URIs**  
  *(The Heart of Linked Data)*
#1: Be Linkable

- Put your information on your website
- Invest in content management
- Consider offering public APIs
- Use Good URLs
  - Readable, Unambiguous, … even in 10 years
- Support caching
- Support content negotiation
- Publish your URL Survival Plan
Think of your data as RDF triples:
  - For each item of interest
  - For each question about that item
  - Supply the answer (if you have it)

About Massachusetts:
  - Governor? Deval Patrick
  - Nickname? “Bay State”
  - Capital? Boston

And Boston? Does it have a nickname? ...
Seen as a “Graph”

- Massachusetts (governor: Deval Patrick)
- Boston (nickname: "Beantown")
- "Bay State" (abbreviation for Massachusetts)
@prefix db: <http://dbpedia.org/resource/> 

db:Massachusetts db:Governor db:Deval_Patrick; 
  db:Nickname "Bay State"; 
  db:Capital db:Boston. 

db:Boston db:Nickname "Beantown".
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:db="http://dbpedia.org/resource/"
    xmlns:db:Governor="http://dbpedia.org/resource/Deval_Patrick"
    xmlns:db:Nickname="http://dbpedia.org/resource/Boston"
>
    <rdf:Description rdf:about="http://dbpedia.org/resource/Massachusetts">
        <db:Governor>
            <rdf:Description rdf:about="http://dbpedia.org/resource/Deval_Patrick" />
        </db:Governor>
        <db:Nickname>Bay State</db:Nickname>
        <db:Capital>
            <rdf:Description rdf:about="http://dbpedia.org/resource/Boston">
                <db:Nickname>Beantown</db:Nickname>
            </rdf:Description>
        </db:Capital>
    </rdf:Description>
</rdf:RDF>
Working with RDF

- Several ways to write down triples
- Lots of software, libraries
- Query language (SPARQL)
- Works over HTTP
- Use Content Negotiation if necessary

But how do you find these triples?
(imagine a web without links)
A first use case for RDF was website metadata

City of Boston website: http://www.cityofboston.gov

Subject: http://www.cityofboston.gov (City of Boston govt site)

Property: When was it created?

Value: 2001-02-01

Possible RDF triple in page
This works great:

- You're given the URL of a Web page
- You do an HTTP GET on that URL
- You get the content
- ... and you get the metadata

This didn't catch on, but....
URLs and URIs

- Web Addresses
  - Uniform Resource Locator (URL)
  - Uniform Resource Identifier (URI)

- Beyond http, Web Addresses:
  - mailto:sandro@hawke.org
  - mid:1276004540.10196.1.camel@waldron

- Maybe everything is a “Resource”? 
  - Maybe everything can have a URI!
#3b: Use Live URIs

- We want identifiers for **things**
  - People, places, governments, companies, products, songs, musicians, concerts, speeches, courses, schools, buildings, walkways, individual plants, species of plants, species of animals, ...

- All the things we might have questions about
- All the questions themselves
- All the answers (when the answer is another thing)
So what URI do we use for Boston itself?

Boston is not a Web Page
- Boston was created in 1630
- cityofboston.gov was created in 2001

Requirements:
- Get data via existing protocols (http, https)
- Don't confused Boston with cityofboston.gov

... a challenge!
Solution 1: Hash URIs

- Use the “fragment” syntax
  - http://www.w3.org/People/Berners-Lee/#Bio
    - Points to a section in the middle of that Web page
- The URI spec gives us a loophole
  - If URI has hash (#), see spec for Content-Type
    - http://www.w3.org/People/Berner-Lee/data
      - Content-Type is RDF, not HTML, so:
        - http://www.w3.org/People/Berners-Lee/data#I
          - identifies Tim, not a section of a Web page
- Oldest approach
  - Used in most W3C Recommendations
Solution #2: Slash URIs

- Uses a loophole in the HTTP spec:
  - GET http://dbpedia.org/resource/Boston
  - Server responds with redirect
    - 303 SEE OTHER
    - LOCATION: http://dbpedia.org/page/Boston
  - GET http://dbpedia.org/page/Boston

- Again, we have two URIs:
  - A URI for the thing itself (Boston)
  - A URI for a Web page/information source about it

- Used for big URI sets (dbpedia, gov't data)
Solution 3?

- TopicPage URLs
  - Use http://www.cityofboston.gov but remember it's indirect
  - Details still need to be worked out
  - This is the kind of URL that Facebook's OpenGraphProtocol uses
The Result?

- Given a URI for Boston
  - http://dbpedia.org/resource/Boston
- An app can find out the mayor's name
  - (assuming that's in the data)
- And the mayors of associated cities
  - (assuming that's in the data)
- And the people associated with those mayors
  - (assuming that's in the data)
- … etc
A Web of Data

- Everyone can publish triples
- Everyone can read the public ones
- Read all the ones you want
- Follow URI links to other interesting data

- As always, it all depends on the data
Related Work

- **RDFS/OWL (Schemas, Ontologies)**
  - Helps in defining sensible properties/classes
  - Helps with automated reasoning about the data
- **SKOS (Controlled Vocabularies)**
  - Simple way to document concepts
- **RIF (Rules)**
  - Translate between vocabularies
- **SPARQL (Query)**
  - Query language/protocol for data in RDF triples
Open Issues

- Expected harvesting diameter?
- Expected inference behavior?
- Finding good sources, vocabularies?
- Establishing backlinks, crosslinks?
- Easy-to-use generalized client?
- Smooth integration with HTML web?
- Business models?
Summary

- **Linked Data**
  - Allows data apps to find/merge data
  - Builds on existing social & Web systems

- **Be Linkable**
  - Use best practices for publishing on the Web

- **Show Your Triples**
  - Publish your data as subject/property/value triples in one or more RDF formats, and maybe SPARQL

- **Use Live URIs**
  - Identify everything with working Hash or Slash URIs