

**An overview of
Semantic Web Languages
and Technologies**

Semantic Web Technologies

- W3C “recommendations”
 - RDF, RDFS, RDFa, OWL, SPARQL, RIF, R2R, etc...
- Common tools and systems -- commercial, free and open sourced
 - Ontology editors, triple stores, reasoners, etc.
- Common ontologies and data sets
 - Foaf, DBpedia, SKOS, PROV, etc.
- Infrastructure systems
 - Search, ontology metadata, linking services
- Non W3C: Schema.org, Freebase, ...

Common KR languages

- [Knowledge representation and reasoning](#) (KR&R) has always been an important part of AI & other disciplines
- Many approaches have been developed, implemented and evolved since the 1960s
- Most were one-offs, used only by their developers
- Starting in the 1990s, there was an interest in developing a common KR language to support knowledge reuse and distributed KB systems
- The Semantic Web languages (e.g., OWL) are a current generation of this idea
 - There are currently no other widely used KR languages

Questions

- Database (DB) vs. knowledge base (KB)?
 - TL;DR: DBs have facts, KBs have general knowledge and (maybe) facts
 - DBs typically have simple schemas (knowledge) and lots of data (facts)
 - KBs have complex schemas (aka ontologies) and may or may not have a lot of instances (data)
- KBs support inference, e.g.,
 $\text{parent}(?x, ?y) \Rightarrow \text{person}(?x), \text{person}(?y), \text{child}(?y, ?x), \text{olldr}(?X, ?y), ?x \neq ?y$
 $\text{Parent}(\text{john}, \text{mary}) \Rightarrow \text{person}(\text{john}), \text{child}(\text{mary}, \text{john}), \dots$

Questions

What's the impact of using different structures to represent data or knowledge?

- Natural language
- Relations vs. graphs vs. objects
- Logic vs. rules vs. procedures
- Neural networks
- Tensors

Questions

What's our “semantic” model for facts and knowledge?

- Classical logic is a common choice
 - $\text{man}(\text{socrates}), \forall x \text{ man}(x) \Rightarrow \text{mortal}(x)$
 - Classical logic has limitations: facts and relations and “rules” are either (always) True or False
- May need to represent and reason with probabilistic or fuzzy facts and knowledge
- May need to handle dynamic facts or knowledge

Semantic Web Technologies

- Basic approach uses classical logic for underlying semantics
 - + Simple, well understood, good reasoning algorithms
 - No probabilities, adding extensions (e.g., for time) adds complexity
- Knowledge represented as a graph
 - + Simple, good tool support
 - May be too simple

Two Semantic Web Notions

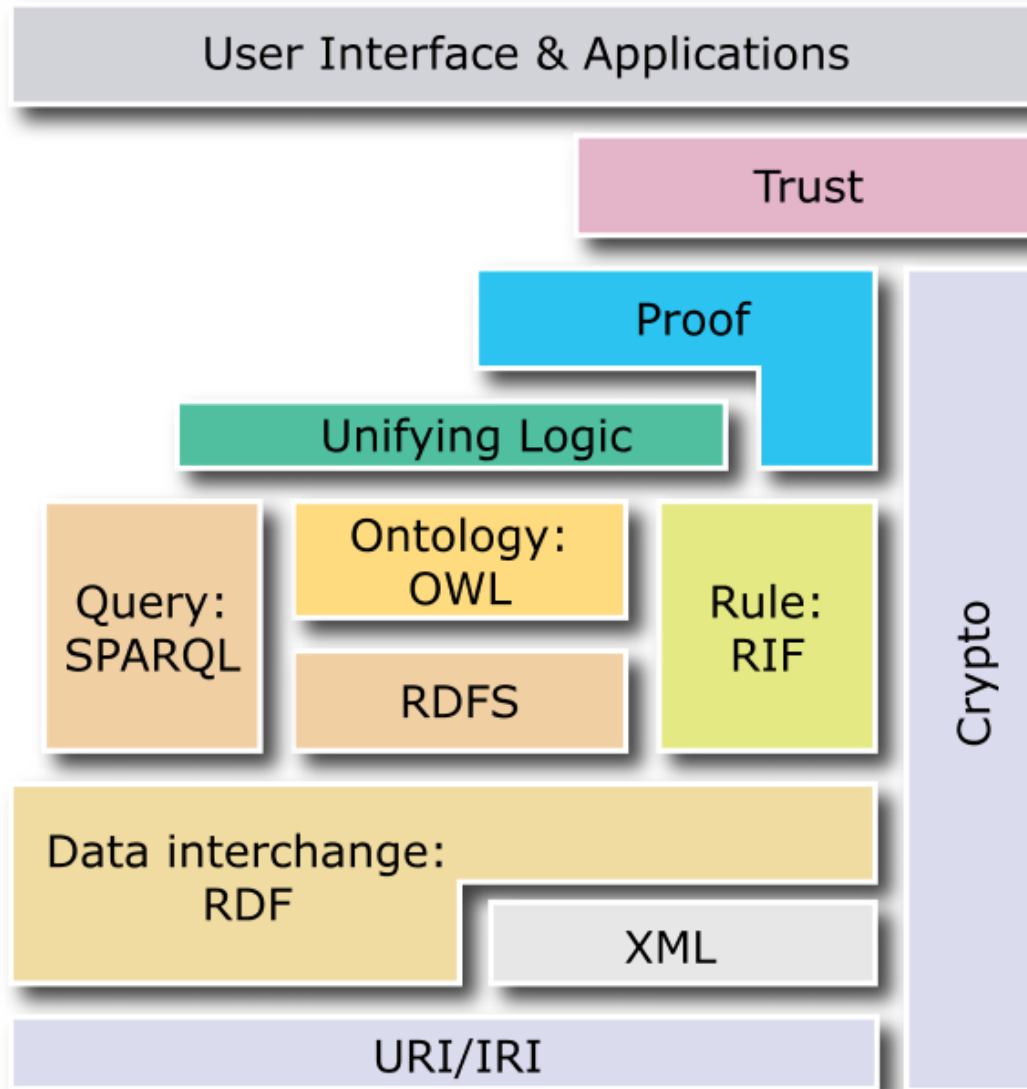
- **The semantic web**

- Idea of a web of machine understandable information
- Agnostic about the technology used to support it
- May involve more AI (e.g., NLP)
- Human end users in the center

- **The Semantic Web**

- The current vision of a semantic web as defined by the W3C community: a web of data
- Using W3C supported standards, i.e., RDF, OWL, SPARQL, XML, RIF, etc.
- By machines for machines with human oriented applications on top

W3C Semantic Web Stack



RDF is the first SW language

XML Encoding

```
<rdf:RDF .....>
  <...>
  <...>
</rdf:RDF>
```

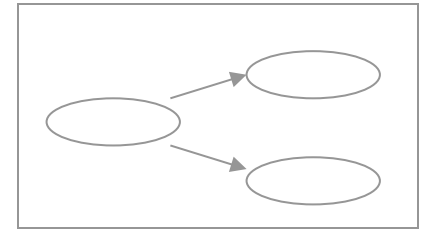
**Good for
Machine
Processing**

JSON Encoding

```
{ "@context": {
  "name": "http://.../name",
  "Person": "http://.../Person"
},
"@type": "Person",
"name": "Markus Lanthaler"
}
```

**RDF
Data Model**

Graph



**Good for
people, viz,
graph DBMS**

Triples

```
stmt(docInst, rdf_type, Document)
stmt(personInst, rdf_type, Person)
stmt(inroomInst, rdf_type, InRoom)
stmt(personInst, holding, docInst)
stmt(inroomInst, person, personInst)
```

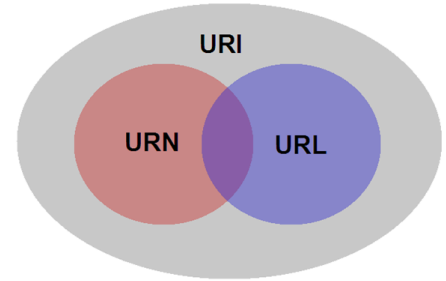
**Good For Reasoning and
Databases**

*RDF is a simple
language for building
graph based
representations*

The RDF Data Model

- An RDF document is an unordered collection of statements, each with a **subject**, **predicate** and **object** (aka **triples**)
- A triple can be thought of as a labelled arc in a graph
- Statements describe properties of web **resources**
- Resources are objects that can be pointed to by a **URI**:
 - a document, a picture, a paragraph on the Web, ...
 - E.g., <http://umbc.edu/~finin/cv.html>
 - a book in the library, a real person (?)
 - isbn://5031-4444-3333
- Properties themselves are also resources (URIs)

URIs are a foundation



- URI = [Uniform Resource Identifier](#)
 - "The generic set of all names/addresses that are short strings that refer to resources"
 - URLs ([Uniform Resource Locators](#)) are a subset of URIs, used for resources that can be *accessed* on the web
- URIs look like URLs, often with fragment identifiers pointing to a document part:
 - `http://foo.com/bar/mumble.html#pitch`

URIs are a foundation

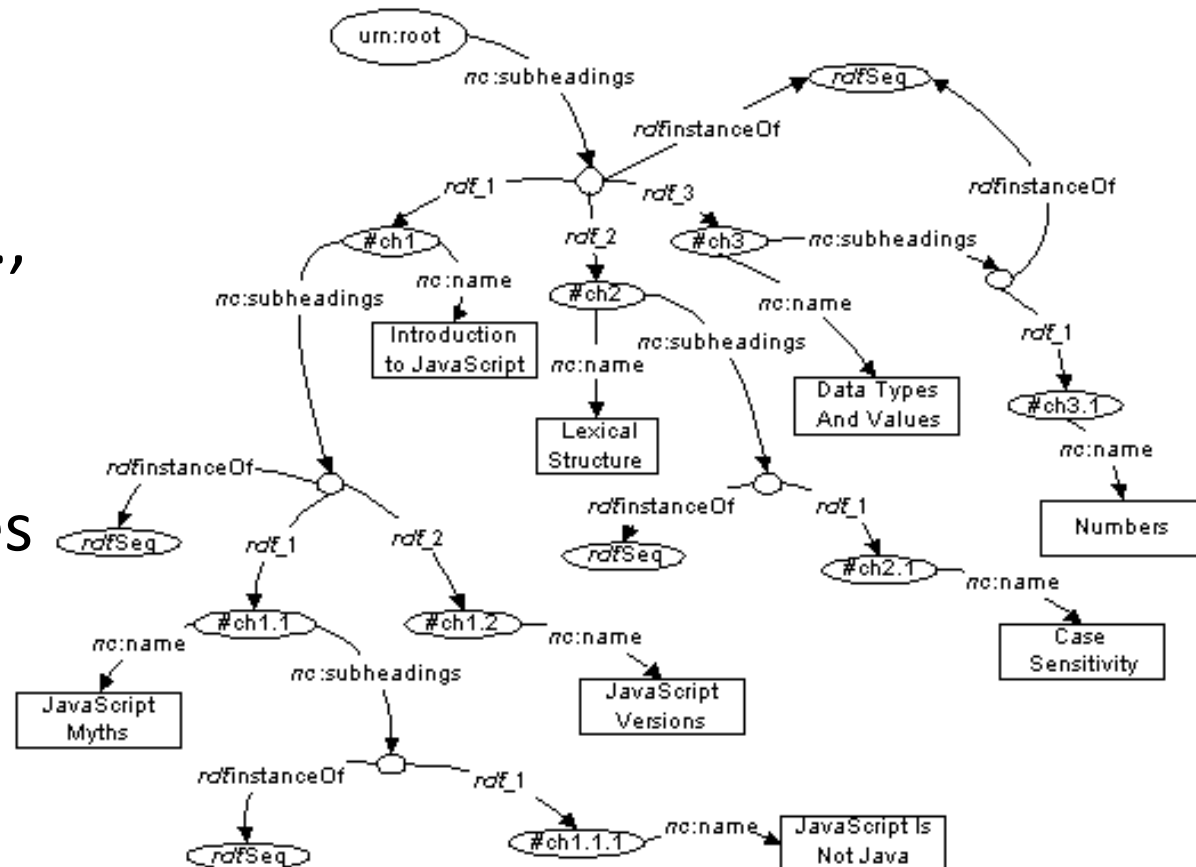
- URIs are unambiguous, unlike natural language terms -- the web provides a global **namespace**
- We can use a URI to **denote** something, e.g., a concept, entity, event or relation
- We usually assume references to the same URI are to the same thing

What does a URI mean?

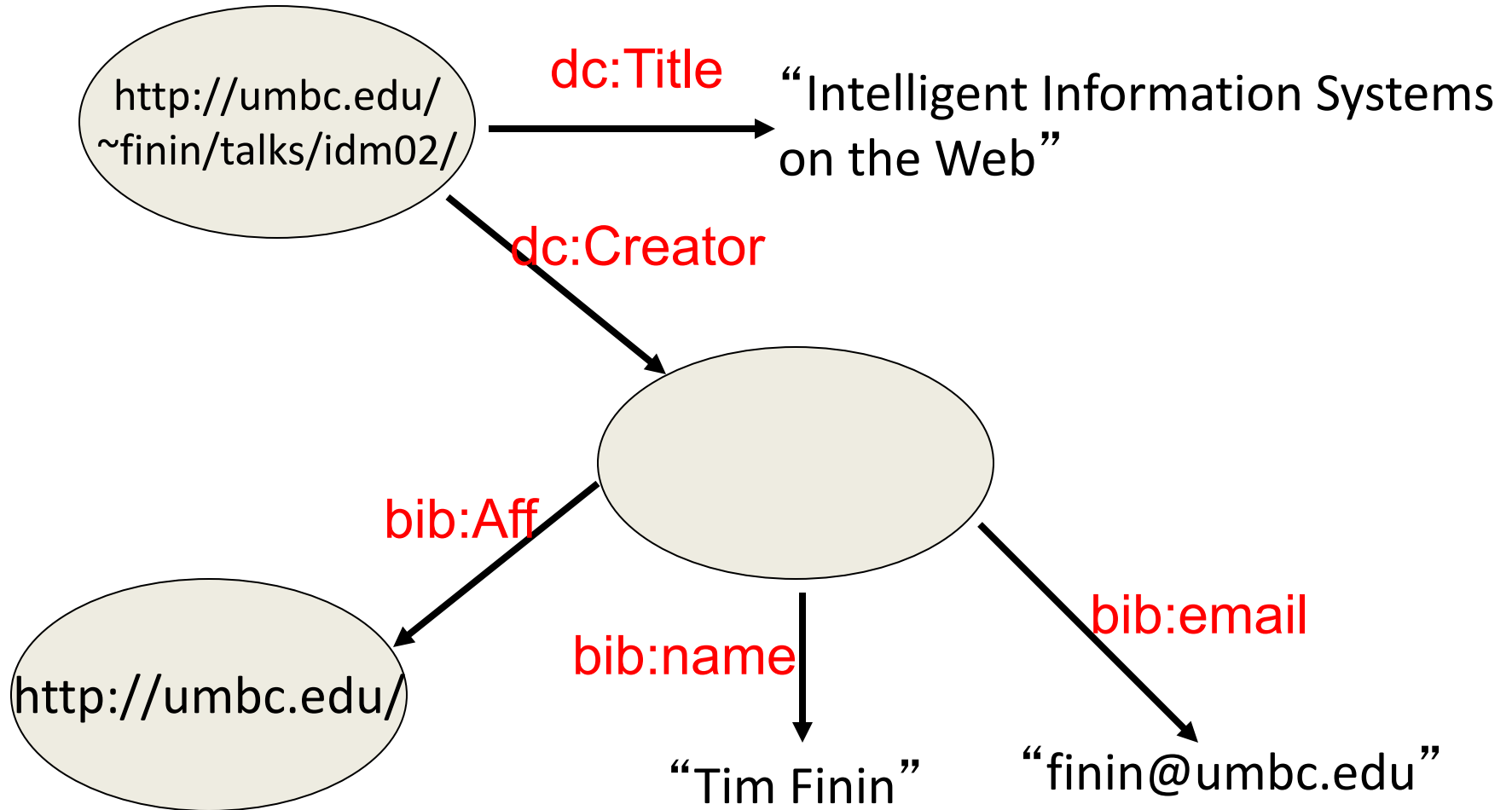
- Sometimes URIs denote a web resource
 - `http://umbc.edu/~finin/finin.jpg` denotes a file
 - We can use RDF to make assertions about the resource, e.g., it's an image and depicts a person with name Tim Finin, ...
- Sometimes concepts in the external world
 - E.g., `http://umbc.edu/` denotes a particular university located in Baltimore
 - This is done by social convention
- Cool URIs don't change
 - <http://www.w3.org/Provider/Style/URI>

The RDF Graph

- An RDF document is an unordered collection of triples
- The subject of one triple can be the object of another
- The result is a directed, labelled graph
- A triple's object can also be a literal, e.g., a string
- Graphs are simpler than relational tables or objects
- This is both a plus and a minus



Simple RDF Example



Serialization

- A graph is an abstract model, we'll need to serialize it as text for many reasons, e.g., display, editing, exchange
- There are several standard RDF serializations, the three most important are: XML, Turtle and ntriples
- Most Semantic Web tools can read or write in any of these serializations

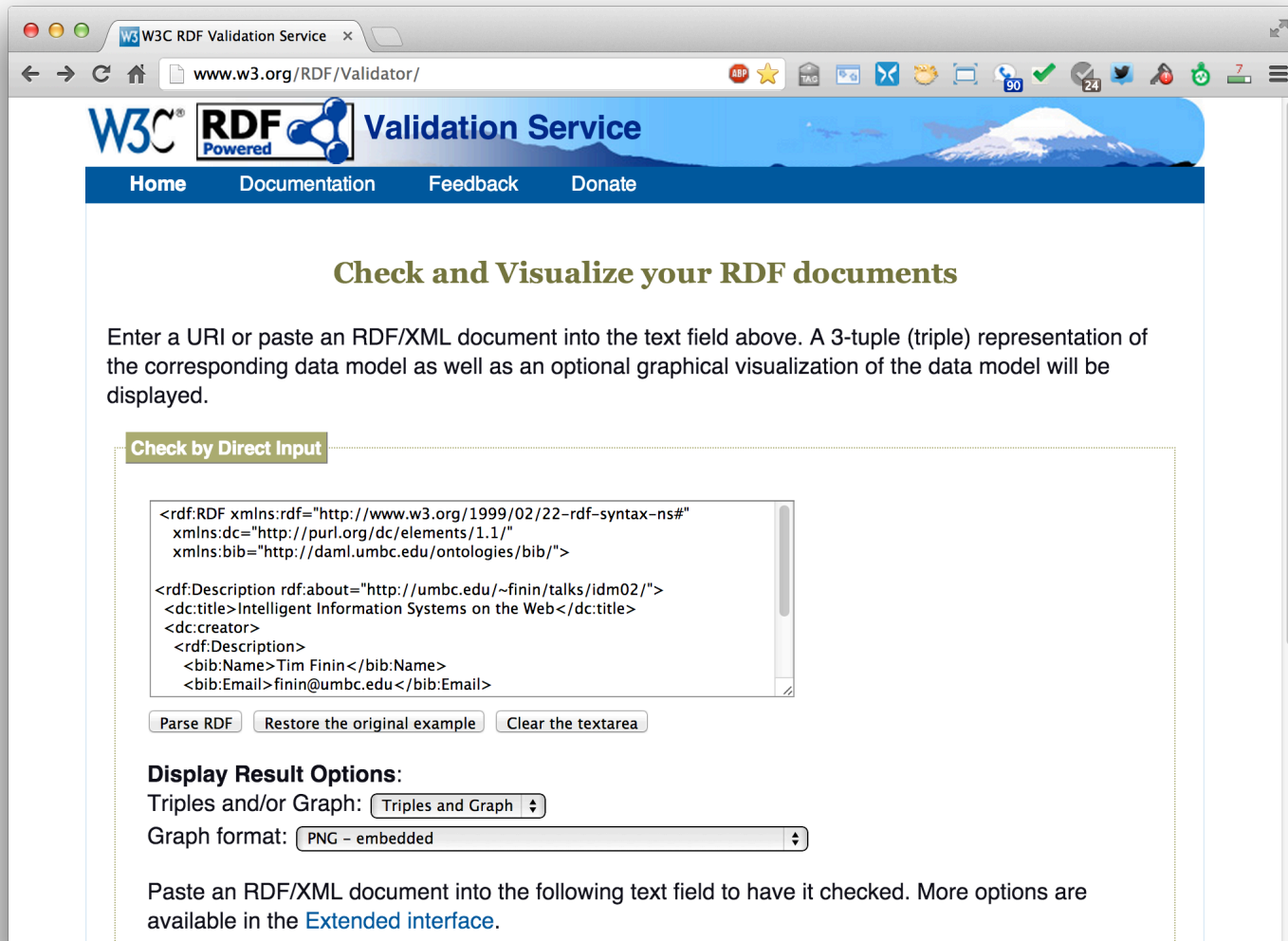
XML encoding for RDF

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:bib="http://daml.umbc.edu/ontologies/bib/">
<rdf:Description rdf:about="http://umbc.edu/~finin/talks/idm02/">
  <dc:title>Intelligent Information Systems on the Web</dc:title>
  <dc:creator>
    <rdf:Description>
      <bib:Name>Tim Finin</bib:Name>
      <bib:Email>finin@umbc.edu</bib:Email>
      <bib:Aff rdf:resource="http://umbc.edu/" />
    </rdf:Description>
  </dc:creator>
</rdf:Description>
</rdf:RDF>
```

Note the prefix declarations

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:bib="http://daml.umbc.edu/ontologies/bib/">
  <rdf:Description rdf:about="http://umbc.edu/~finin/talks/idm02/">
    <dc:title>Intelligent Information Systems on the Web</dc:title>
    <dc:creator>
      <rdf:Description>
        <bib:Name>Tim Finin</bib:Name>
        <bib:Email>finin@umbc.edu</bib:Email>
        <bib:Aff rdf:resource="http://umbc.edu/" />
      </rdf:Description>
    </dc:creator>
  </rdf:Description>
</rdf:RDF>
```

An RDF validation service



The screenshot shows a web browser window with the address bar displaying `www.w3.org/RDF/Validator/`. The page header features the W3C logo, the text "RDF Powered", and "Validation Service". A navigation bar includes links for "Home", "Documentation", "Feedback", and "Donate". The main heading is "Check and Visualize your RDF documents". Below this, a paragraph explains that users can enter a URI or paste an RDF/XML document to get a 3-tuple representation and an optional graphical visualization. A section titled "Check by Direct Input" contains a text area with the following RDF/XML code:

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:bib="http://daml.umbc.edu/ontologies/bib/">

  <rdf:Description rdf:about="http://umbc.edu/~finin/talks/idm02/">
    <dc:title>Intelligent Information Systems on the Web</dc:title>
    <dc:creator>
      <rdf:Description>
        <bib:Name>Tim Finin</bib:Name>
        <bib:Email>finin@umbc.edu</bib:Email>
      </rdf:Description>
    </dc:creator>
  </rdf:Description>
</rdf:RDF>
```

Below the text area are three buttons: "Parse RDF", "Restore the original example", and "Clear the textarea". Under the heading "Display Result Options:", there are two dropdown menus: "Triples and/or Graph:" set to "Triples and Graph" and "Graph format:" set to "PNG - embedded". At the bottom, a paragraph instructs users to paste an RDF/XML document into the text field and mentions more options in the "Extended interface".

<http://www.w3.org/RDF/Validator/>

Easy to convert between serializations

- Most software tools can read and write different serializations
- [rdf2rdf](#) is a simple handy utility for converting from one RDF serialization to another
- [Any23](#) is an open source library, web service and command line tool that extracts structured data in RDF format from a variety of Web documents

N-triple representation

RDF can be encoded as a set of **triples**

<subject> <predicate> <object> .

<http://umbc.edu/~finin/talks/idm02/> <http://purl.org/dc/elements/1.1/title>
"Intelligent Information Systems on the Web" .

<http://umbc.edu/~finin/talks/idm02/> <http://purl.org/dc/elements/1.1/creator>
_:node17i6ht38ux1 .

_:node17i6ht38ux1 <http://daml.umbc.edu/ontologies/bib/Name> "Tim Finin" .

_:node17i6ht38ux1 <http://daml.umbc.edu/ontologies/bib/Email> "finin@umbc.edu" .

_:node17i6ht38ux1 <http://daml.umbc.edu/ontologies/bib/Aff> <http://umbc.edu/> .

N3/Turtle notation for RDF

- N3 is a compact notation for RDF that is easier for people to read, write and edit
- It's just [syntactic sugar](#)
- Aka Notation 3, developed by TBL himself
- But, XML is largely unreadable and even harder to write
- Turtle is a W3C standard that covers most of N3

Turtle Example

@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

@prefix dc: <http://purl.org/dc/elements/1.1/> .

@prefix bib: <http://daml.umbc.edu/ontologies/bib/> .

<<http://umbc.edu/~finin/talks/idm02/>>

dc:title "Intelligent Information Systems on the Web" ;

dc:creator

[bib:Name "Tim Finin";

bib:Email "finin@umbc.edu"

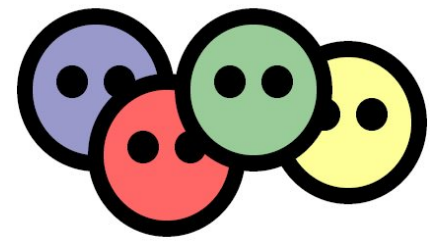
bib:Aff: "<http://umbc.edu/>"] .

Triple Notes

- **RDF triples have one of two forms:**
 - <URI> <URI> <URI>
 - <URI> <URI> <quoted string>
- **Triples are also easily mapped into logic**
 - <subject> <predicate> <object> becoming:
 - <predicate>(<subject>,<object>)
 - With type(<S>,<O>) becoming <O>(<S>)
 - Example:
 - subclass(man,person)
 - sex(man,male)
 - domain(sex,animal)
 - man(adam)
 - age(adam,100)
- **Triples are easily stored and managed in DBMS**
 - Flat nature of a triple a good match for relational DBs

*; Note: we're not
; showing the actual
; URIs for clarity*

A usecase: FOAF



- FOAF (Friend of a Friend) is a simple ontology to describe people and their properties and social networks
 - See the foaf project page: <http://www.foaf-project.org/>
- In 2008 we crawled the web and found > 1M valid RDF FOAF files.
 - Most were from <http://liveJournal.com/> blogging system which encoded basic user in FOAF
 - Apache currently use FOAF for open source committers
 - See <http://xmlns.com/foaf/spec/> for the vocabulary specification

```
<foaf:Person>
  <foaf:name>Tim Finin</foaf:name>
  <foaf:mbox_sha1sum>2410...37262c252e</foaf:mbox_sha1sum>
  <foaf:homepage rdf:resource="http://umbc.edu/~finin/" />
  <foaf:img rdf:resource="http://umbc.edu/~finin/images/passport.gif" />
</foaf:Person>
```

FOAF Vocabulary

Basics

[Agent](#)
[Person](#)
[name](#)
[nick](#)
[title](#)
[homepage](#)
[mbox](#)
[mbox_sha1sum](#)
[img](#)
[depiction](#) ([depicts](#))
[surname](#)
[family_name](#)
[givenname](#)
[firstName](#)

Personal Info

[weblog](#)
[knows](#)
[interest](#)
[currentProject](#)
[pastProject](#)
[plan](#)
[based_near](#)
[workplaceHomepage](#)
[workInfoHomepage](#)
[schoolHomepage](#)
[topic_interest](#)
[publications](#)
[geekcode](#)
[myersBriggs](#)
[dnaChecksum](#)

Documents & Images

[Document](#)
[Image](#)
[PersonalProfileDocument](#)
[topic](#) ([page](#))
[primaryTopic](#)
[tipjar](#)
[sha1](#)
[made](#) ([maker](#))
[thumbnail](#)
[logo](#)

Online Accts

[OnlineAccount](#)
[OnlineChatAccount](#)
[OnlineEcommerceAccount](#)
[OnlineGamingAccount](#)
[holdsAccount](#)
[accountServiceHomepage](#)
[accountName](#)
[icqChatID](#)
[msnChatID](#)
[aimChatID](#)
[jabberID](#)
[yahooChatID](#)

Projects & Groups

[Project](#) [Organization](#)
[Group](#) [member](#)
[membershipClass](#) [fundedBy](#)
[theme](#)

FOAF: why RDF? Extensibility!

- FOAF vocabulary has 50+ basic terms for simple facts about people
- FOAF files can use other RDF terms too: RSS, MusicBrainz, Dublin Core, Wordnet, Creative Commons, blood types, starsigns, ...
- RDF gives freedom of independent extension
 - OWL provides fancier data-merging facilities
- Freedom to *say what you like*, using any RDF markup you want, and have RDF crawlers merge your FOAF documents with other's and know when you're talking about the same entities

No free lunch!

- Must plan for lies, mischief, mistakes, stale data, slander, idiosyncratic terms
- Dataset is out of control, distributed, dynamic
- Importance of knowing who-said-what
 - Anyone can describe anyone
 - We must record data **provenance**
 - Modeling and reasoning about trust is critical
- Legal, privacy and etiquette issues emerge
- Welcome to the real world

More RDF Vocabulary

- RDF has terms for describing lists, bags, sequences, simple datatypes, etc.
- RDF is a “pure” graph representation language
 - Nodes and edges are simple objects
 - Both have identifiers that are URIs
- Suppose we want to associate a probability with an edge, e.g.,
 - (:flipper rdf:type :mammal) :probability 0.9
 - (:flipper rdf:type :fish) :probability 0.1

Property graphs?

- RDF is a “pure” graph model with only labeled nodes and edges
- Many popular graph databases implement property graphs (e.g., [Neo4j](#))
- Nodes and edges can have one or more properties, whose values are literals or maybe lists of literals
- Results in a more compact graph
- But, as we’ll see, introduces some limitations

More RDF Vocabulary

- RDF also can describe triples through reification
- Enabling statements about statements

:flipper rdf:type :mammal .

- All non-literals have to be URIs
- RDF uses prefixes for readability
- We can specify what a null prefix means
- If we don't it means "in this file"
- <https://prefix.cc/> is one service for looking up prefixes

More RDF Vocabulary

- RDF also can describe triples through reification
- Enabling statements about statements

```
:flipper rdf:type :mammal .  
_:s1 rdf:type rdf:Statement .  
_:s1 rdf:subject :flipper .  
_:s1 rdf:predicate :type .  
_:s1 rdf:object :mammal .  
_:s1 :probability 0.9
```

- The underscore prefix is special
- It introduces *blank nodes*
- We'll talk about this in more detail later
- For now, think of it as introducing “a new, nameless thing”

More RDF Vocabulary

- RDF also can describe triples through reification
- Enabling statements about statements

```
:flipper rdf:type :mammal .
```

```
_:s1 a rdf:Statement;  
    rdf:subject :flipper;  
    rdf:predicate :type;  
    rdf:object :mammal;  
    :probability 0.9 .
```

More RDF Vocabulary

- RDF also can describe triples through reification
- Enabling statements about statements

:john bdi:believes _:s.

_:s rdf:type rdf:Statement.

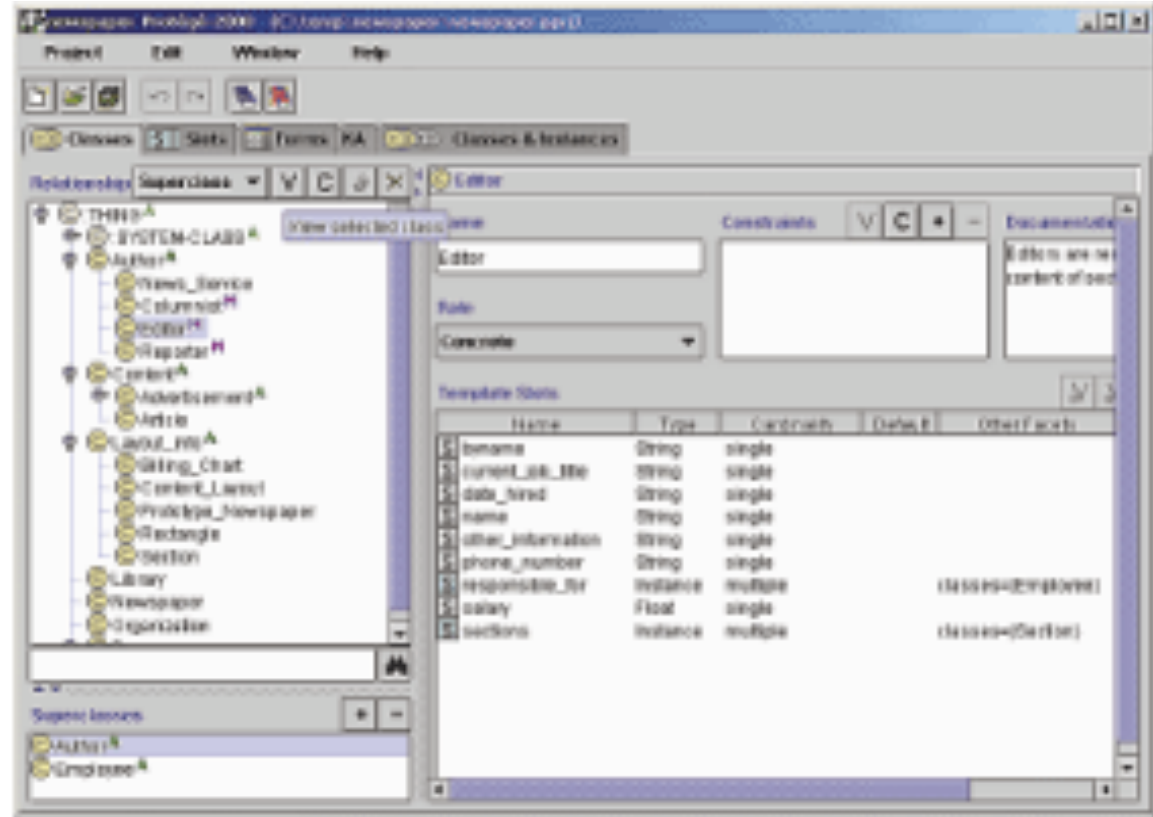
_:s rdf:subject <http://ex.com/catalog/widgetX>.

_:s rdf:predicate cat:salePrice .

_:s rdf:object "19.95" .

RDF Schema (RDFS)

- **RDF Schema adds taxonomies for classes & properties**
 - subClass and subProperty
- **and some metadata.**
 - domain and range constraints on properties
- **Several widely used KB tools can import and export in RDFS**



Stanford Protégé KB editor

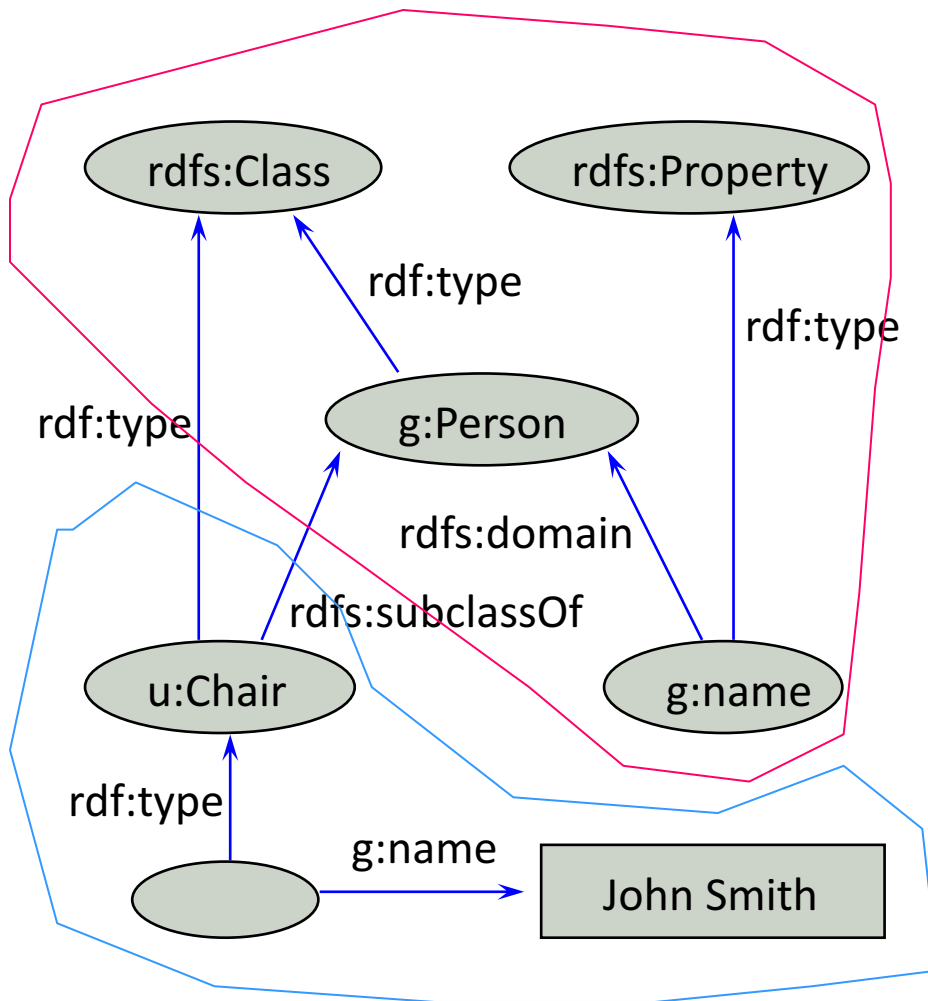
- Java, open sourced
- extensible, lots of plug-ins
- provides reasoning & server capabilities

RDFS Vocabulary

RDFS introduces the following terms and gives each a meaning w.r.t. the rdf data model

- Terms for classes
 - [rdfs:Class](#)
 - [rdfs:subClassOf](#)
- Terms for properties
 - [rdfs:domain](#)
 - [rdfs:range](#)
 - [rdfs:subPropertyOf](#)
- Special classes
 - [rdfs:Resource](#)
 - [rdfs:Literal](#)
 - [rdfs:Datatype](#)
- Terms for collections
 - [rdfs:member](#)
 - [rdfs:Container](#)
 - [rdfs:ContainerMembershipProperty](#)
- Special properties
 - [rdfs:comment](#)
 - [rdfs:seeAlso](#)
 - [rdfs:isDefinedBy](#)
 - [rdfs:label](#)

RDF and RDF Schema



```
<rdfs:Property rdf:ID="name">
  <rdfs:domain rdf:resource="Person">
</rdfs:Property>
```

```
<rdfs:Class rdf:ID="Chair">
  <rdfs:subClassOf rdf:resource=
    "http://schema.org/gen#Person">
</rdfs:Class>
```

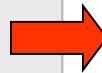
```
<rdf:RDF
  xmlns:g="http://schema.org/gen"
  xmlns:u="http://schema.org/univ">
  <u:Chair rdf:ID="john">
    <g:name>John Smith</g:name>
  </u:Chair>
</rdf:RDF>
```

RDFS supports simple inferences



- An RDF ontology plus some RDF statements may imply additional RDF statements
- Not true of XML data
- Note that this is **part of the data model** and not of the accessing or processing code

```
@prefix rdfs: <http://www...>.
@prefix : <...genesis.n3>.
:parent rdfs:domain :Person;
        rdfs:range :Person.
:mother
  rdfs:subProperty parent;
  rdfs:domain :Woman.
:eve :mother :cain.
```



```
:parent a rdf:Property.
:Person a rdf:Class.
:Woman rdfs:subClassOf Person.
:mother a rdf:Property.
:eve a :Person;
      a :Woman;
      :parent :cain.
:cain a :Person.
```

N3 example

Here's how you declare a namespace.

<> is an alias for the URI of this document.

This document is referring to "this document".

```
@prefix rdf: <http://www.w3.org/1999/02/22/rdf-schema#>.
@prefix : <#>.
<> rdf:type rdf:Class.
:Person a rdf:Class.
:Woman a rdf:Class; rdfs:subClassOf :Person.
:eve a :Woman; :age "100".
:sister a rdf:Property; rdfs:domain :Person; rdfs:range :Woman.
:eve :sister [a :Woman; :age 98].
:eve :believe { :eve :age "100" }.
[is :spouse of [is :sister of :eve]] :age 99.
:eve.:sister.:spouse :age 99.
```

"person is a class". The "a" syntax is sugar for rdf:type.

"Woman is a class and a subclass of person". Note the ; syntax.

"eve is a woman whose age is 100."

"sister is a property from person to woman"

"eve has a sister who is 98"

"eve believes that her age is 100". The braces are used to group the triplets.

"the spouse of the sister of eve is 99".

"the spouse of the sister of eve is 99".

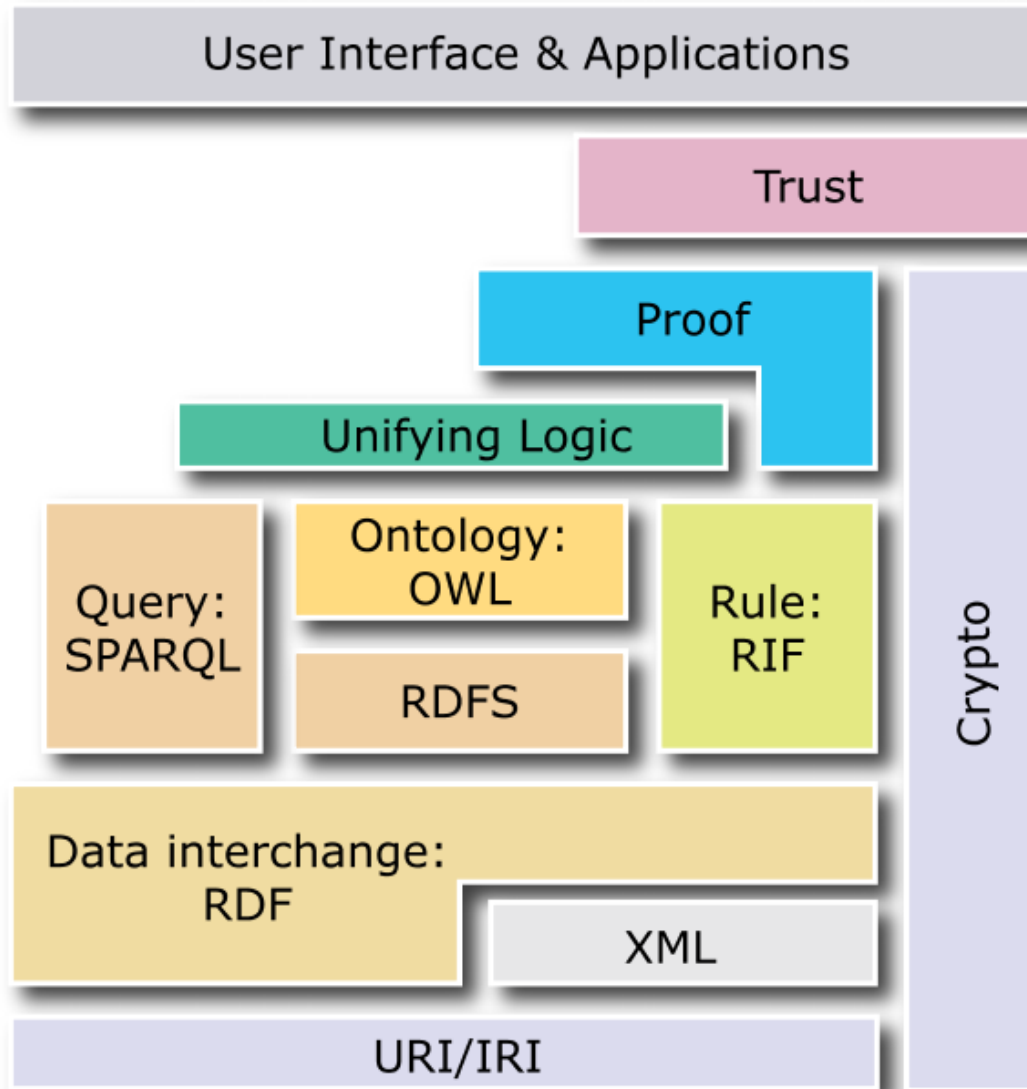
Is RDF(S) better than XML?

Q: For a specific application, should I use XML or RDF?

A: It depends...

- XML's model is
 - a tree, i.e., a strong hierarchy
 - applications may rely on hierarchy position
 - relatively simple syntax and structure
 - not easy to *combine* trees
- RDF's model is
 - a *loose* collections of relations
 - applications may do database-like search
 - not easy to recover hierarchy
 - easy to combine relations in one big collection
 - great for the integration of heterogeneous information

W3C Semantic Web Stack



Problems with RDFS

- RDFS **too weak** to describe resources in detail, e.g.
 - No *localised range and domain* constraints
Can't say that the range of hasChild is person when applied to persons and dog when applied to dogs
 - No *existence/cardinality* constraints
Can't say that all *instances* of person have a mother that is also a person, or that persons have exactly two parents
 - No *transitive, inverse or symmetrical* properties
Can't say isPartOf is a transitive property, hasPart is the inverse of isPartOf or touches is symmetrical
- We need RDF terms providing these and other features.

W3C's Web Ontology Language (OWL)

- DARPA project, DAML+OIL, begat OWL
- OWL released as W3C recommendation 2/10/04
- See the [W3C OWL pages](#) for overview, guide, specification, test cases, etc.
- Three layers of OWL are defined of decreasing levels of complexity and expressiveness
 - **OWL Full** is the whole thing
 - **OWL DL** (Description Logic) introduces restrictions
 - **OWL Lite** is an entry level language intended to be easy to understand and implement
- Owl 2 became a W3C recommendation in 2009

OWL $\leftrightarrow$ RDF

- An OWL ontology is a set of RDF statements
 - OWL defines semantics for certain statements
 - Does **NOT** restrict what can be said; documents can include arbitrary RDF
 - But no OWL semantics for non-OWL statements
- Adds capabilities common to [description logics](#), e.g., cardinality constraints, defined classes, equivalence, disjoint classes, etc.
- Supports ontologies as objects (e.g., importing, versioning, ...)
- A complete OWL reasoning is significantly more complex than a complete RDFS reasoner.

Embedding Semantic Data in HTML

- Embedding semantic data in HTML allows documents to be understood by people and machines
 - RDFa is a ‘standard’ for embedding RDF in HTML as tag attributes
 - JSON-LD is a ‘standard’ for embedding RDF in a simple json-compatible serialization
- Facebook looks for embedded RDFa statements using its opengraph (og) vocabulary
- Bestbuy embeds product info in RDFa

Detecting semantic data via a browser

Apple Pie by Grandma Ople x RDFa: http://allrecipes.com/ x

allrecipes.com search by ingredient recipes » videos » menus » holidays »

RECIPE BOX SHOPPING LISTS MENU PLANNER ALLRECIPES MAGAZINE Go Pro! or Join f

938 Photos

Apple Pie by Grandma Ople

READY IN **1 1/2** hrs

Sponsor Get the magazine

★★★★★ Read Reviews (5755)

Pinit 14K+ Tweet 103 +1 969

Recipe by MOSHASHAMA

"This was my grandmother's apple pie recipe. I have never seen another one quite like it. It will always be my favorite and has won me several first place prizes in local competitions. I hope it becomes one of your favorites as well!"

See how to make this recipe!

+ Recipe Box + Shopping List + Menu Email Print

Ingredients Edit and Save

Original recipe makes 1 - 9 inch pie Change Servings

1 recipe pastry for a 9 inch 1/2 cup white sugar

Watch video tips and tricks

In Season

Football Wings, dip Get the re

Healthy Find good love to ea

Get Back Find recip goals.

Related Videos

Detecting semantic data via a browser

Plugin shows up if RDFa detected. Shows statements if clicked.
<http://bit.ly/gturtle>

Plugin shows up if Schema.org statements detected. Shows statements if clicked.
<http://bit.ly/sdotoch>

allrecipes.com

search by i

menus » holidays »

Go Pro! or Join fo

Pie by Grand

the magazine

★ ★ ★ Read Review

K+

MOSHASMAMA

my grandmother's apple pie recipe. I have never another one quite like it. It will always be my favorite and has won me several first place prizes in local competitions. I hope it becomes one of your favorites as well!"

See how to make this recipe!

+ Recipe Box + Shopping List + Menu Email Print

Ingredients [Edit and Save](#)

Original recipe makes 1 - 9 inch pie [Change Servings](#)

1 recipe pastry for a 9 inch 1/2 cup white sugar

Watch video tips and tricks

Related Videos

Semantic Data Browser/Query

The screenshot shows a web browser window with the address bar displaying `dbpedia.org/page/Alan_Turing`. The page title is "About: Alan Turing". The DBpedia logo is visible in the top left, and navigation links for "Browse using", "Formats", "Faceted Browser", and "Sparql Endpoint" are in the top right. The main content area features the heading "About: Alan Turing" and a description of Alan Turing as a scientist. Below this is a table with two columns: "Property" and "Value". The first row shows the property `dbo:abstract` and its value, which is a detailed biographical paragraph about Alan Turing.

DBpedia

Browse using Formats Faceted Browser Sparql Endpoint

About: Alan Turing

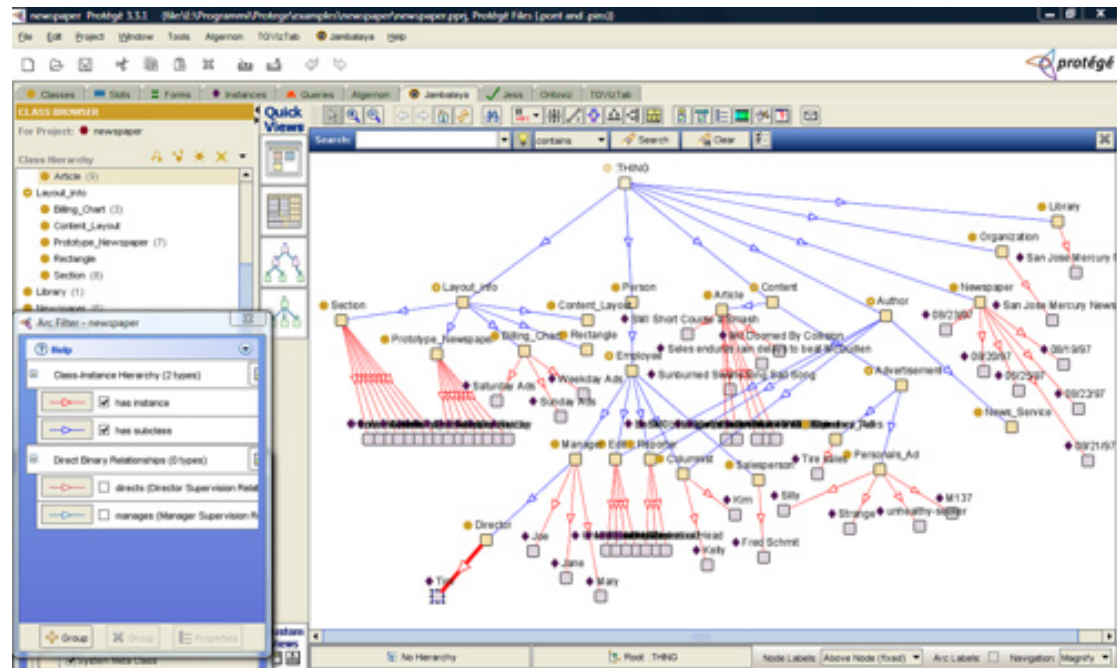
An Entity of Type : [scientist](#), from Named Graph : <http://dbpedia.org>, within Data Space : [dbpedia.org](#)

Alan Mathison Turing, OBE, FRS (/ˈtjʊərɪŋ/; 23 June 1912 – 7 June 1954) was a British pioneering computer scientist, mathematician, logician, cryptanalyst, theoretical biologist, and marathon and ultra distance runner. He was highly influential in the development of computer science, providing a formalisation of the concepts of algorithm and computation with the Turing machine, which can be considered a model of a general purpose computer.

Property	Value
dbo:abstract	Alan Mathison Turing, OBE, FRS (/ˈtjʊərɪŋ/; 23 June 1912 – 7 June 1954) was a British pioneer biologist, and marathon and ultra distance runner. He was highly influential in the development and computation with the Turing machine, which can be considered a model of a general purpose computer science and artificial intelligence. During the Second World War, Turing worked for the codebreaking centre. For a time he led Hut 8, the section responsible for German naval cryptos

Ontology Editor

- There are a number of editors available for creating and editing ontologies and data
- We recommend using [Protégé](#), a java-based free system developed at Stanford
 - Good support for reasoning
 - Lots of plugins



Triple Stores

- A triple store is a database for RDF triples
- It usually has a native API and often accepts SPARQL queries
- It might do reasoning, either in an *eager* manner (as triples are loaded) or *on demand* (to answer queries), etc
- Some stores focus on scalability and others on flexibility and features
- We'll look at several, including [Sesame](#), Apache [Jena](#) and [stardog](#)

Frameworks and Libraries

- There are frameworks, libraries and packages for most programming languages
- [Jena](#) is a very comprehensive Java framework originally developed by HP and now Apache
- Others are available for Python, Ruby, C#, Perl, PHP, Prolog, Lisp, etc.

Conclusion

- There's quite a bit of technology needed to support the Semantic Web
- This has been a brief tour
- We'll cycle back on these and explore them in more detail
- And give you a chance to use and experiment with them