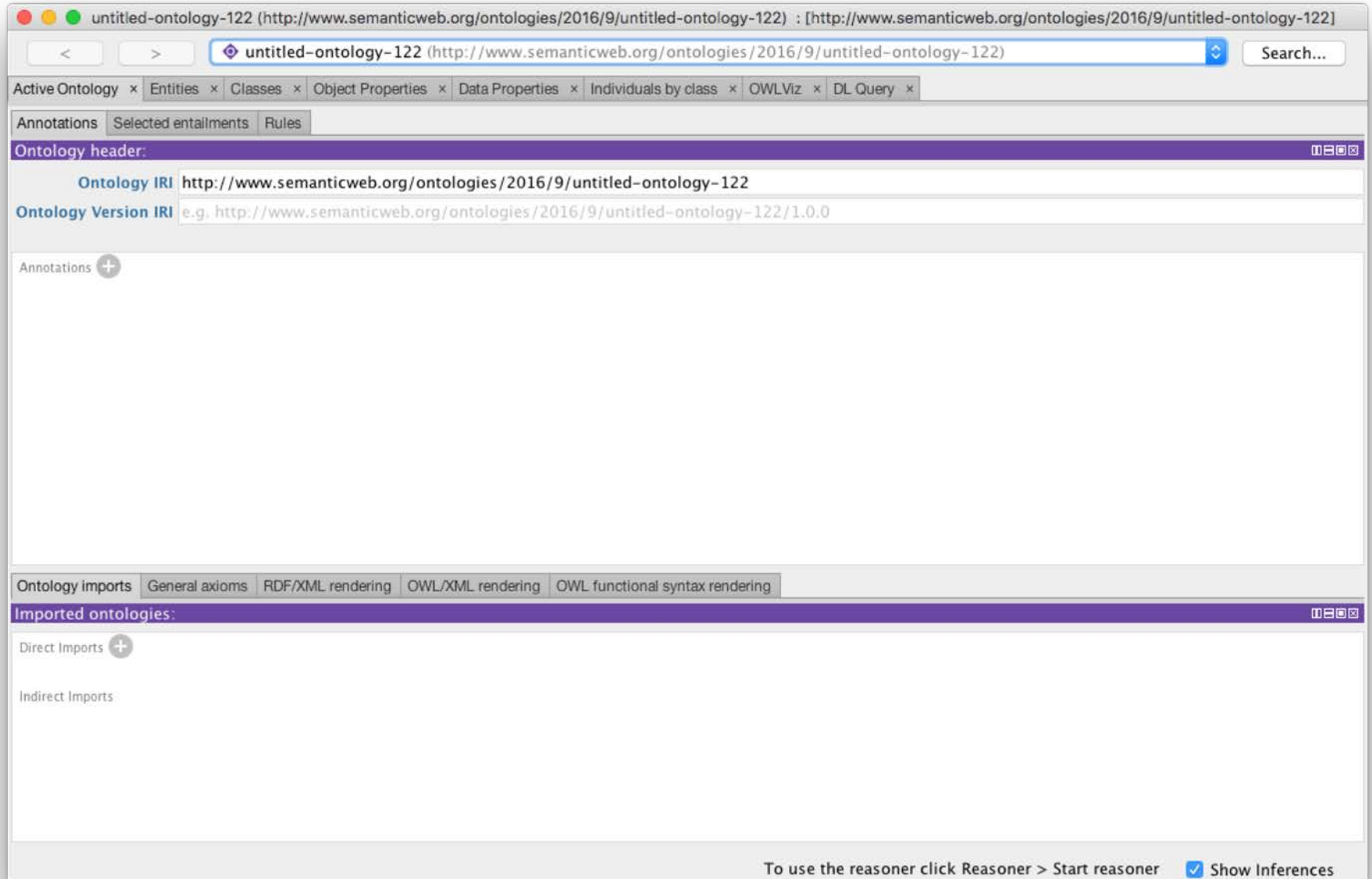


Ontology Editors

IDEs for Ontologies

- Some people use simple text editors
 - Working with XML serialization will drive you crazy
 - Using Turtle or an abstract syntax works well
- Others prefer an IDE
 - Good IDEs include support for reasoning, visualization, and more
- Protégé is a very popular IDE
 - From Stanford, free, lots of plugins
- TopQuadrant Composer is also good
 - Feature rich but expensive (\$600 for a single license)

Protégé 5.1



Protégé 5.2

- <http://protege.stanford.edu/>
- Free, open source ontology editor and KB framework
- Predates OWL, still supports earlier Frames representation
- In Java, extensible, large community of users
- Desktop and Web versions
 - Works will under Linux, Mac OS X and Windows

Desktop Protégé

The screenshot displays the Desktop Protégé interface for the ontology 'peeps' (http://ebiq.org/ontologies/peeps/). The interface is divided into several panes:

- Top Bar:** Shows the ontology name and URL, a search bar, and tabs for 'Active Ontology', 'Entities', 'Individuals by class', 'DL Query', 'Individual Hierarchy Tab', and 'OWLviz'.
- Left Pane:** Contains a 'Class hierarchy' view showing a tree structure: `owl:Thing` (parent) → `Person` (child) → `Adult` (child) → `Man` (child, highlighted). Other children of `Person` include `Boy`, `Minor`, `Boy`, and `Woman`.
- Right Pane:** Displays details for the selected class `Man` (http://ebiq.org/ontologies/peeps/#Man). It includes:
 - Class Annotations:** `rdfs:label` with value 'Male person' and `rdfs:comment` with value 'A Man is defined as a person with a has_sex value equal to "male"'. Each annotation has edit, delete, and help icons.
 - Description: Man:** Shows the class definition: `Equivalent To` `Person and (hasSex value "male")`.
 - SubClass Of:** Lists `Person` and `(hasSex value "male")`.
 - General class axioms:** Shows `SubClass Of (Anonymous Ancestor)` with two axioms: `hasParent exactly 1 Man` and `hasParent exactly 1 Woman`.
- Bottom Left Pane:** Contains a 'Datatypes' view showing a tree structure: `owl:topDataProperty` (parent) → `owl:topObjectProperty` (child).
- Bottom Right:** A status bar with the text 'To use the reasoner click Reasoner > Start reasoner' and a checked checkbox for 'Show Inferences'.

Web Protégé

The screenshot shows the WebProtégé web application interface. The browser window has a single tab titled "WebProtégé" with the URL "webprotege.stanford.edu/#Edit:...". The application header includes the Protégé logo, a "Project" dropdown menu, a "Share" button, a user profile "Tim Finin" with a dropdown arrow, and a "Help" dropdown menu. Below the header is a breadcrumb trail: "WebProtégé" > "UMBC691PeepsExample". A horizontal menu contains tabs for "Classes", "Properties", "Individuals", "Notes and Discussions", "Changes By Entity", and "Project Dashboard". A bar below the menu offers "Add content to this tab" and "Add tab" options.

The main workspace is divided into three panels:

- Classes Panel:** Displays a tree structure under "owl:Thing" with "Person" and "Sex" as subclasses. It includes "Create", "Delete", and "Watch Br" buttons.
- Class description for Person Panel:** Contains fields for "Display name" (Person) and "IRI" (http://webprotege.stanford.edu/RCcwwdIsdJMKb). It also features an "Annotations" section with a table showing "rdfs:label" for "Person" in the default language, and a "Properties" section with an input field for adding new properties.
- Discussions for Person Panel:** Includes a "Post new topic..." button.

YAS: Yet Another Syntax

- Neither OWL's official abstract syntax nor XML serialization is easy to read or use
- Protégé uses the Manchester syntax
- Simpler and more compact: “some” and “only”, not “someValuesFrom” and “allValuesFrom”
- A W3C recommendation (<http://bit.ly/manSyn>), used in the OWL 2 Primer (<http://bit.ly/OWL2Pri>)

Class: man

Annotations: rdfs:label "man"

EquivalentTo: adult and male and person

Manchester OWL syntax

OWL	DL Symbol	Manchester OWL Syntax Keyword	Example
someValuesFrom	\exists	some	hasChild some Man
allValuesFrom	\forall	only	hasSibling only Woman
hasValue	\ni	value	hasCountryOfOrigin value England
minCardinality	\geq	min	hasChild min 3
cardinality	$=$	exactly	hasChild exactly 3
maxCardinality	\leq	max	hasChild max 3

Manchester OWL syntax

OWL	DL Symbol	Manchester OWL Syntax Keyword	Example
intersectionOf	\sqcap	and	Doctor and Female
unionOf	\sqcup	or	Man or Woman
complementOf	\neg	not	not Child

Example

```
Person and
  hasChild some
    (Person and
      (hasChild only Man) and
        (hasChild some Person) )
```

The set of people who have at least one child that has some children that are only men (i.e., grandparents that only have grandsons)

Data values and datatypes

- Data values typed or untyped (e.g., int, boolean, float)
- Constants with or w/o type, e.g.: hasAge value "21"^^long
- Use datatype names as classes: hasAge some int
- XSD facets, e.g.: Person and hasAge some int[>= 65]

XSD facet	Meaning
< x, <= x	less than, less than or equal to x (more info)
> x, >= x	greater than, greater than or equal to x (more info)
length x	For strings, the number of characters must be equal to x (more info)
maxLength x	For strings, the number of characters must be less than or equal to x (more info)
minLength x	For strings, the number of characters must be greater than or equal to x (more info)
pattern regexp	The lexical representation of the value must match the regular expression, regexp (more info)
totalDigits x	Number can be expressed in x characters (more info)
fractionDigits x	Part of the number to the right of the decimal place can be expressed in x characters (more info)

Demonstration

- We'll use Protégé OWL v5.2 to implement a tiny ontology for people
- Start by downloading and installing Protégé 5.2 (You will need Java)
- You may want to install Graphviz
- Configure Protégé
 - E.g., select a reasoner to use (e.g., HermiT)

A basic workflow

- Think about usecases
- Preliminaries
 - Choose namespace URL, import other ontologies used
- Identify and define classes
 - Place in hierarchy, add **axioms** and run reasoner to check for errors or omissions
- Identify and define properties
 - Place in hierarchy, add **axioms**, run reasoner
- Add individuals & reasoner to check for problems
- Add comments and labels
- Export in desired formats, maybe upload to Web

More workflow steps

- Use [OOPS](#) to find common ontology pitfalls
- Link concepts (and individuals) to common ontologies (e.g., DBpedia, Freebase, foaf)
 - Use owl:sameAs
- Generate visualizations
- Produce documentation
- Develop examples with your use case(s)
- Encode data, describe in [VoID](#) (Vocabulary of Interlinked Datasets), add to LOD cloud

Demonstration

Use Protégé OWL (v5.2) to build a simple ontology for people based on the following

- People have just one sex that's either *male* or *female*, an integer age, and two parents, one male, one female
- A person's grandparent is the parent of their parent
- Every person is either a man or a woman but not both
- A man is defined as any person whose sex is male and a woman as any person whose sex is female
- A boy is defined as a person whose sex is male and whose age is less than 18, a girl is ...
- A person is either an adult or (age >18), minor (age <18), ...

Test cases

All Different people

Alice F

Bob M

Carol F

Don M

Edith F

Pat ?

Other people

Frank M

Gwen F

Some possible test cases

- Alice parent Bob . Bob parent Carol
 - Alice grandparent Carol
- Alice parent Bob . Alice parent Don.
 - Contradiction
- Alice parent Bob . Pat parent Bob
 - Pat a female
- Alice parent Bob . Gwen parent Bob .
 - Alice owl:sameAs Gwen