

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 5.5

The screenshot displays the Protégé 5.5 web interface. At the top, the browser address bar shows the URL `http://www.semanticweb.org/ontologies/2016/9/untitled-ontology-122`. Below the address bar, a search bar contains the text `untitled-ontology-122`. The main interface is divided into several sections:

- Active Ontology:** A tabbed interface with options: `Active Ontology`, `Entities`, `Classes`, `Object Properties`, `Data Properties`, `Individuals by class`, `OWL Viz`, and `DL Query`.
- Annotations:** A tabbed interface with options: `Annotations`, `Selected entailments`, and `Rules`.
- Ontology header:** A section with two fields:
 - Ontology IRI:** `http://www.semanticweb.org/ontologies/2016/9/untitled-ontology-122`
 - Ontology Version IRI:** `e.g. http://www.semanticweb.org/ontologies/2016/9/untitled-ontology-122/1.0.0`
- Annotations:** A large empty area with a `+` icon in the top left corner.
- Ontology imports:** A tabbed interface with options: `General axioms`, `RDF/XML rendering`, `OWL/XML rendering`, and `OWL functional syntax rendering`.
- Imported ontologies:** A section with two fields:
 - Direct Imports:** A field with a `+` icon.
 - Indirect Imports:** A field.

At the bottom of the interface, a status bar contains the text: `To use the reasoner click Reasoner > Start reasoner` and a checked checkbox labeled `Show Inferences`.

Protégé 5.5

- <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
 - Requires Java Runtime Environment
- [Desktop](#) and [Web](#) versions
 - Works will under Linux, Mac OS X and Windows

Desktop Protégé

The screenshot displays the Protégé desktop application interface for the ontology 'peeps' (http://ebiq.org/ontologies/peeps/). The main window shows the 'Class hierarchy' tab, where the 'Man' class is selected. The left sidebar contains a tree view of the ontology, showing the hierarchy: owl:Thing > Person > Adult > Man > Boy > Minor > Boy > Woman. The right pane shows the 'Annotations: Man' tab, which lists the following annotations:

- Annotations:** Man
 - Annotations:**
 - rdfs:label**: Male person
 - rdfs:comment**: A Man is defined as a person with a has_sex value equal to "male"
 - Description: Man**
 - Equivalent To**: Person and (hasSex value "male")
 - SubClass Of**: General class axioms
 - SubClass Of (Anonymous Ancestor)**
 - hasParent exactly 1 Man
 - hasParent exactly 1 Woman

The bottom status bar indicates: To use the reasoner click Reasoner > Start reasoner ☒ Show Inferences.

Web Protégé

The screenshot displays the WebProtégé web application interface. The browser window title is "WebProtégé" and the address bar shows "webprotege.stanford.edu/#Edit:...". The interface includes a top navigation bar with the Protégé logo, a "Project" dropdown, "Share", "Tim Finin" dropdown, and "Help" dropdown. Below this is a tab bar with "WebProtégé" and "UMBC691PeepsExample". A secondary tab bar contains "Classes", "Properties", "Individuals", "Notes and Discussions", "Changes By Entity", and "Project Dashboard". A toolbar at the top right offers "Add content to this tab" and "Add tab".

The main content area is divided into three panels:

- Classes:** A sidebar on the left showing a hierarchy starting with "owl:Thing", which includes "Person" and "Sex".
- Class description for Person:** The central panel for editing the "Person" class. It includes:
 - Display name:** "Person"
 - IRI:** "http://webprotege.stanford.edu/RCcwwdIsdJMKB"
 - Annotations:** A table with one row:

rdfs:label	Person	lang
------------	--------	------
 - Properties:** A table with one row:

Enter property	Ent	lang
----------------	-----	------
- Discussions for Person:** A sidebar on the right with 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 1

How can we define a class that is people who have children and all of them are male?

Example: People with just boys

How can we define a class that is people who have children and all of them are male?

Define as the union of three classes

1. Person
2. Things that have children
3. Things where all of their children are mail

Example: People with just boys

How can we define a class that is people who have children and all of them are male?

Define as the union of three classes

1. Person
2. Things that have children
3. Things where all of their children are male

An **owl:someValuesFrom**
restriction on hasChild
property

An **owl:allValuesFrom**
restriction on hasChild
property

Example: People with just boys

How can we define a class that is people who have children and all of them are male?

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

Example 2

```
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 a Pjb)

Data values and datatypes

- Data values typed or untyped (e.g., int, boolean, float)
- Constants w/ 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]
- Ranges: Person and hasAge some int[>= 18, <= 30]

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 v5.5 to implement a tiny ontology for people
- Start by downloading and installing Protégé 5.5
(You will need the JRE installed)
- 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
Ontology Pitfall Scanner detect many common pitfalls introduced when developing ontologies
- 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/HW4

Use Protégé OWL (v5.5) 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