

# Lab: Ontology Engineering (2) – Protégé

Camilo Thorne

Room 00.012

Institut für Maschinelle Sprachverarbeitung  
Universität Stuttgart

+49 (0) 711 685-81369

[camilo.thorne@ims.uni-stuttgart.de](mailto:camilo.thorne@ims.uni-stuttgart.de)

Semantic Web, SS 2017

(based on slides by W. Kessler)

# Outline

- 1 Protégé
- 2 Classes
- 3 Properties
- 4 Instances
- 5 Other Views
- 6 Reasoner

- We really don't want to write RDF/RDFS/OWL by hand.
- Thankfully, there exist tools to help us!
- Protégé<sup>1</sup> is a well-known and powerful ontology editor.
- Protégé is free, open source and implemented in Java.
- Protégé is maintained by the Stanford Center for Biomedical Informatics Research and the community.



---

<sup>1</sup><http://protege.stanford.edu/>

- The different parts of the ontology you are looking at are shown in different tabs (views).
- You choose which tabs to show/hide in “Window” – “Tabs”.
- Protégé allows plugins to add their own tabs.

# Outline

- 1 Protégé
- 2 **Classes**
- 3 Properties
- 4 Instances
- 5 Other Views
- 6 Reasoner



# The Classes View (2)

- The "Classes" view contains all information about the classes in the ontology.
- The "Class Hierarchy" windows shows the taxonomy of the ontology. You can add new sibling classes or subclasses with the buttons at the left.
- "Usage" displays where the selected class is being referred to.
- In the "Description" window you can add definitions by clicking on the "+" symbol.
  - You can chose a class from the "class hierarchy" tab.
  - You can use the GUI-creators for simple restrictions on object properties or data properties.
  - You can use the class expression editor with a special syntax
- You can edit definitions by clicking on the "@" symbol.

# The Classes View (3) – Class Definitions

**Equivalent classes** Equivalent classes defined by name or restrictions ("necessary and sufficient criteria").

**Superclasses** A list of classes defined by name or restrictions, that this class is a subclass of ("necessary criteria").

**Inherited anonymous classes** A list of conditions that have been inherited from superclasses.

**Members** All instances of this class.

**Keys** A value of a "key" unambiguously identifies a particular instance, no two instances of the class can have the same value in the "key" property.

**Disjoint classes** Disjoint classes defined by name or restrictions.

**Disjoint union of** If A is a disjoint union of B1 and B2, then everything that is a member of A and not a member of B1 is a member of B2.



# Protégé Syntax for Class Expressions

```
Pizza and (not (hasTopping some FishTopping))  
and (not (hasTopping some MeatTopping))  
and hasTopping min 3
```

Name	Protégé Syntax	OWL Syntax
Existential restrictions	<code>some</code>	<code>owl:someValuesFrom</code>
Universal restrictions	<code>only</code>	<code>owl:allValuesFrom</code>
hasValue restriction	<code>value</code>	<code>owl:hasValue</code>
Intersection	<code>and</code>	<code>owl:intersectionOf</code>
Union	<code>or</code>	<code>owl:unionOf</code>
Complement	<code>not</code>	<code>owl:complementOf</code>
Minimum cardinality	<code>min</code>	<code>owl:minCardinality</code>
Maximum cardinality	<code>max</code>	<code>owl:maxCardinality</code>
Cardinality	<code>exactly</code>	<code>owl:cardinality</code>

# Outline

- 1 Protégé
- 2 Classes
- 3 Properties**
- 4 Instances
- 5 Other Views
- 6 Reasoner

# The Properties Views (1)

The screenshot displays the Protégé ontology editor interface. The browser window title is "pizza (http://www.co-ode.org/ontologies/pizza/pizza.owl)". The main menu includes File, Edit, View, Reasoner, Tools, Refactor, Window, and Help. The "Object Properties" tab is active, showing a tree view of properties under "topObjectProperty":

- hasCountryOfOrigin
- hasIngredient
  - hasBase
  - hasTopping
  - hasSpiciness
- isIngredientOf
  - isBaseOf
  - isToppingOf

The "Annotations" panel for "hasTopping" shows a comment: "Note that hasTopping is inverse functional because isToppingOf is functional"@en.

The "Characteristics" panel for "hasTopping" shows the following options:

- Functional
- Inverse functional
- Transitive
- Symmetric
- Asymmetric
- Reflexive
- Irreflexive

The "Description" panel for "hasTopping" shows the following information:

- Domains (intersection): Pizza
- Ranges (intersection): PizzaTopping
- Equivalent object properties: (empty)
- Super properties: hasIngredient
- Inverse properties: isToppingOf
- Disjoint properties: (empty)
- Property chains: (empty)

At the bottom of the window, it states: "No Reasoner set. Select a reasoner from the Reasoner menu" and "Show Inferences" is checked.

## The Properties Views (2)

- Object properties and data properties are separated into the views "Object Properties" and "Data properties".
- The "Data/Object Property Hierarchy" windows shows the taxonomy of the properties. You can add new sibling properties or subproperties with the buttons at the left.
- "Usage" displays where this property is being referred to.
- In the "Description" window you can add domain, range, superproperties and inverse properties by clicking on the "+" symbol.
- In the "Characteristics" window a property can be defined as transitive, (a)symmetric, (ir)reflexive or (inverse) functional<sup>2</sup>.

---

<sup>2</sup>If a property is functional, for a given individual, there can be at most one other individual that is related to this individual via the functional property.

# Outline

- 1 Protégé
- 2 Classes
- 3 Properties
- 4 Instances**
- 5 Other Views
- 6 Reasoner

# The Instances View (1)

The screenshot displays the Protégé software interface for the 'pizza' ontology. The main window title is 'pizza (http://www.co-ode.org/ontologies/pizza/pizza.owl)'. The menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, and Help. The toolbar shows navigation and search icons. The interface is divided into several panes:

- Class hierarchy (inferred):** Shows a tree structure starting with 'Thing', followed by 'DomainConcept', 'Country', 'Food', 'IceCream', 'Pizza', and various subclasses like 'CheeseyPi', 'Interestin', 'MeatyPiz', 'NamedPiz', 'NonVeget', 'Realltalia', 'SpicyPiz', 'ThinAndC', 'Vegetaria', 'Vegetaria', 'Vegetaria', 'PizzaBase', 'PizzaToppin', and 'ValuePartition'.
- Members list (inferred):** Lists instances for the selected class 'Germany': America, England, France, Germany, and Italy.
- Annotations: Germany:** A panel for viewing and editing annotations for the selected instance.
- Description: Germany:** Shows the class hierarchy for the instance, including 'Country' and 'Thing'.
- Property assertions: Germany:** A panel for viewing and editing property assertions for the instance, showing 'Object property assertions', 'Data property assertions', 'Negative object property assertions', and 'Negative data property assertions'.

At the bottom of the window, a status bar indicates 'No Reasoner set. Select a reasoner from the Reasoner menu' and a checked box for 'Show Inferences'.

# The Instances View (2)

- Instances are managed in the view "Individuals".
- The "Class Hierarchy" window shows the taxonomy of the classes.
- When you select a class, the "Members list" window will show all individuals that are instances of this class.
- "Usage" displays where the selected individual is being referred to.
- In the "Description" window you can add types (which classes the individual is an instance of), same and different individuals.
- In the "Property assertions" window you can add object and data properties for the individual (e.g., hasPrice 5.0, ...).

# Outline

- 1 Protégé
- 2 Classes
- 3 Properties
- 4 Instances
- 5 Other Views**
- 6 Reasoner



# Protégé's other Views (1)

**Active Ontology** Shows an overview of the active ontology, including metrics on its contents, annotations about the ontology as a whole, and other imported ontologies (if any imports exist).

**Entities** An overview of the whole ontology containing classes, properties and instances.

**Annotation Properties** Properties that are only “comments” for humans to describe the ontology and its parts and not used in reasoning.

## Protégé's other Views (2)

**OWLViz** Shows a graphical representation of part of the class hierarchy when you click on a class. Requires installation of Graphviz before anything will be visible.

**OntoGraf** Visualization of the classes of the ontology.

**DLQuery** An arbitrary class description in Description Logics can be entered and the reasoner is queried for the sub/super classes, inferred members, etc. Requires the reasoner to run beforehand.

**SPARQL Query** Allows the execution of arbitrary SPARQL queries on the ontology.

# Outline

- 1 Protégé
- 2 Classes
- 3 Properties
- 4 Instances
- 5 Other Views
- 6 Reasoner

- A reasoner can test whether or not one class is a subclass of another class.
  - This results in an inferred class hierarchy.
- A reasoner can also test for consistency, based on the description (restrictions) of a class the reasoner can check whether or not it is possible for the class to have any instances.
  - Inconsistent classes are subclasses of **Nothing**.
- Protégé has different reasoners, select one in the menu "Reasoner" and use "Start Reasoner" to start the test.

# The Pizza Ontology in Protégé's Reasoner

The screenshot shows the Protégé ontology editor interface. The main window title is "pizza (http://www.co-ode.org/ontologies/pizza/pizza.owl)". The menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, and Help. The toolbar shows navigation and search icons. The "Active Ontology" tab is selected, and the "Class hierarchy" pane on the left shows a tree structure. The "Annotations" pane on the right displays the following information for the class "CheeseVegetableTopping":

- Annotations:** comment: "This class will be inconsistent. This is because we have given it 2 disjoint parents, which means it could never have any members (as nothing can simultaneously be a CheeseTopping and a VegetableTopping). NB Called ProbelnconsistentTopping in the ProtegeOWL Tutorial."@en
- Label:** "CoberturaDeQueijoComVegetais"@pt
- Description:** CheeseVegetableTopping
- Equivalent classes:** Nothing
- Superclasses:** CheeseTopping, VegetableTopping, American, AmericanHot, AnchoviesTopping, ArtichokeTopping, AsparagusTopping, Cajun, CajunSpiceTopping, CaperTopping, Capricciosa, ...

At the bottom of the window, the status bar indicates "Reasoner active" and "Show Inferences" is checked.

# The Inferred Class Hierarchy Tab

- The "inferred class hierarchy" tab has a yellow background, as well as all classes added to the description by the reasoner.
- Classes that have changed their superclasses in the hierarchy are marked in blue (e.g., `Caprina` is now a subclass of `VegetarianPizza`).
- Inconsistent classes are marked in red and are subclasses of `Nothing` (e.g., `IceCream`).
- Equivalent classes are marked with  $\equiv$  (e.g., `SpicyPizza` and `SpicyPizzaEquivalent`).