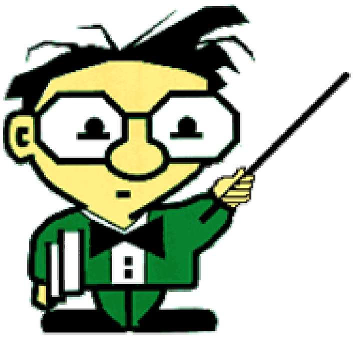
2nd International Semantic Web Conference (ISWC2003)

Tutorial:
Creating Semantic Web (OWL) Ontologies with Protégé

Holger Knublauch, Mark A. Musen, Natasha F. Noy

Sanibel Island, Florida, USA, October 20-23th, 2003

Creating Semantic Web (OWL) Ontologies with Protégé



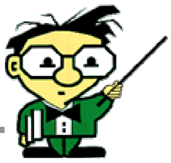
Holger Knublauch
holger@smi.stanford.edu

Mark A. Musen
musen@smi.stanford.edu

Natasha F. Noy
noy@smi.stanford.edu

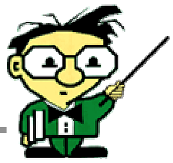
Stanford Medical Informatics

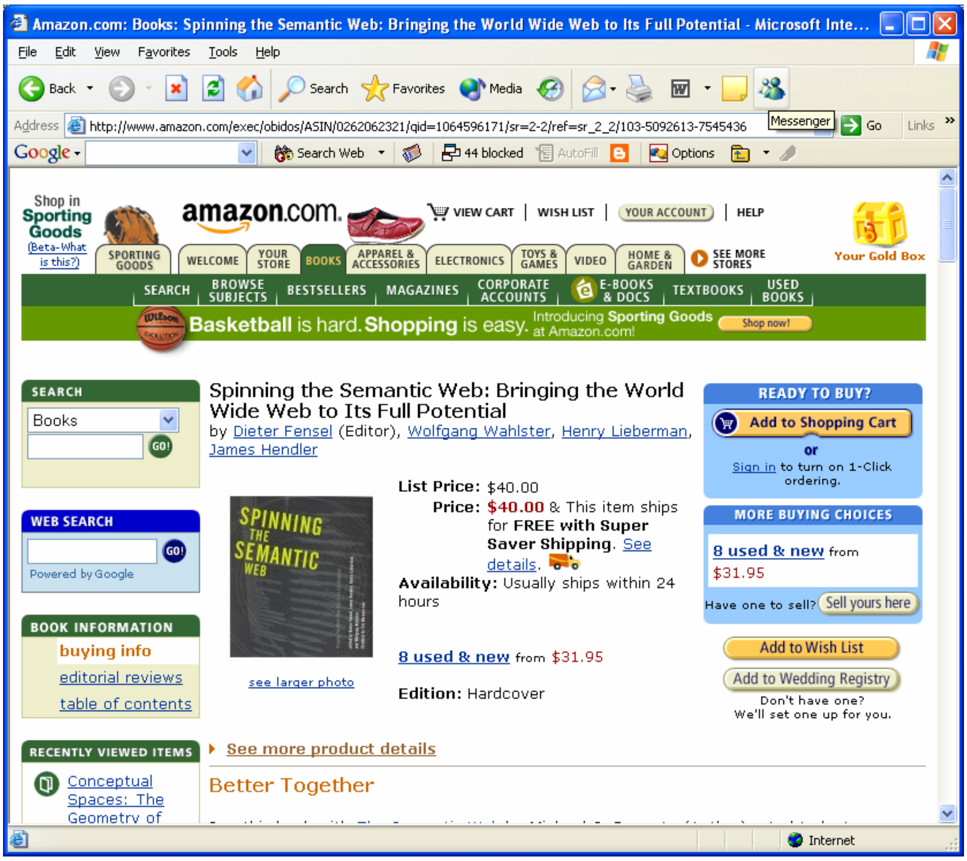
Overview



- Ontologies and the Semantic Web
  - The Semantic Web Vision
  - What is an ontology?
  - The Web Ontology Language (OWL)
- Editing OWL Ontologies with Protégé
  - General Introduction to Protégé
  - Editing the basic elements of OWL (Classes, Properties, Individuals)
  - Editing class expressions
  - Using a classifier and PromptDiff
- Advanced Protégé
  - Visual ontology editing with the GraphWidget and ezOWL
  - Metaclasses
  - Understanding the OWL Plugin
  - Developing plugins and applications with the Protégé/OWL API
- Current limitations and outlook

Semantic Web / Motivation (1)




Amazon.com: Books: Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential - Microsoft Inte...
File Edit View Favorites Tools Help
Back Search Favorites Media
Messenger
Address http://www.amazon.com/exec/obidos/ASIN/0262062321/qid=1064596171/sr=2-2/ref=sr_2_2/103-5092613-7545436
Go Links
Google Search Web 44 blocked AutoFill Options
Shop in Sporting Goods (Beta-What is this?)
amazon.com
VIEW CART | WISH LIST | YOUR ACCOUNT | HELP
SPORTING GOODS WELCOME YOUR STORE BOOKS APPAREL & ACCESSORIES ELECTRONICS TOYS & GAMES VIDEO HOME & GARDEN SEE MORE STORES
Your Gold Box
SEARCH BROWSE SUBJECTS BESTSELLERS MAGAZINES CORPORATE ACCOUNTS E-BOOKS & DOCS TEXTBOOKS USED BOOKS
Basketball is hard. Shopping is easy. Introducing Sporting Goods at Amazon.com! Shop now!
SEARCH Books GO!
WEB SEARCH GO! Powered by Google
BOOK INFORMATION buying info editorial reviews table of contents
RECENTLY VIEWED ITEMS Conceptual Spaces: The Geometry of
SPINNING THE SEMANTIC WEB
see larger photo
READY TO BUY? Add to Shopping Cart or Sign in to turn on 1-Click ordering.
MORE BUYING CHOICES 8 used & new from $31.95
Have one to sell? Sell yours here
Add to Wish List
Add to Wedding Registry
Don't have one? We'll set one up for you.
Internet

Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential
by Dieter Fensel (Editor), Wolfgang Wahlster, Henry Lieberman, James Hendler

List Price: $40.00
Price: $40.00 & This item ships for **FREE with Super Saver Shipping**. See details.
Availability: Usually ships within 24 hours

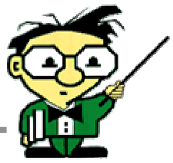
8 used & new from $31.95

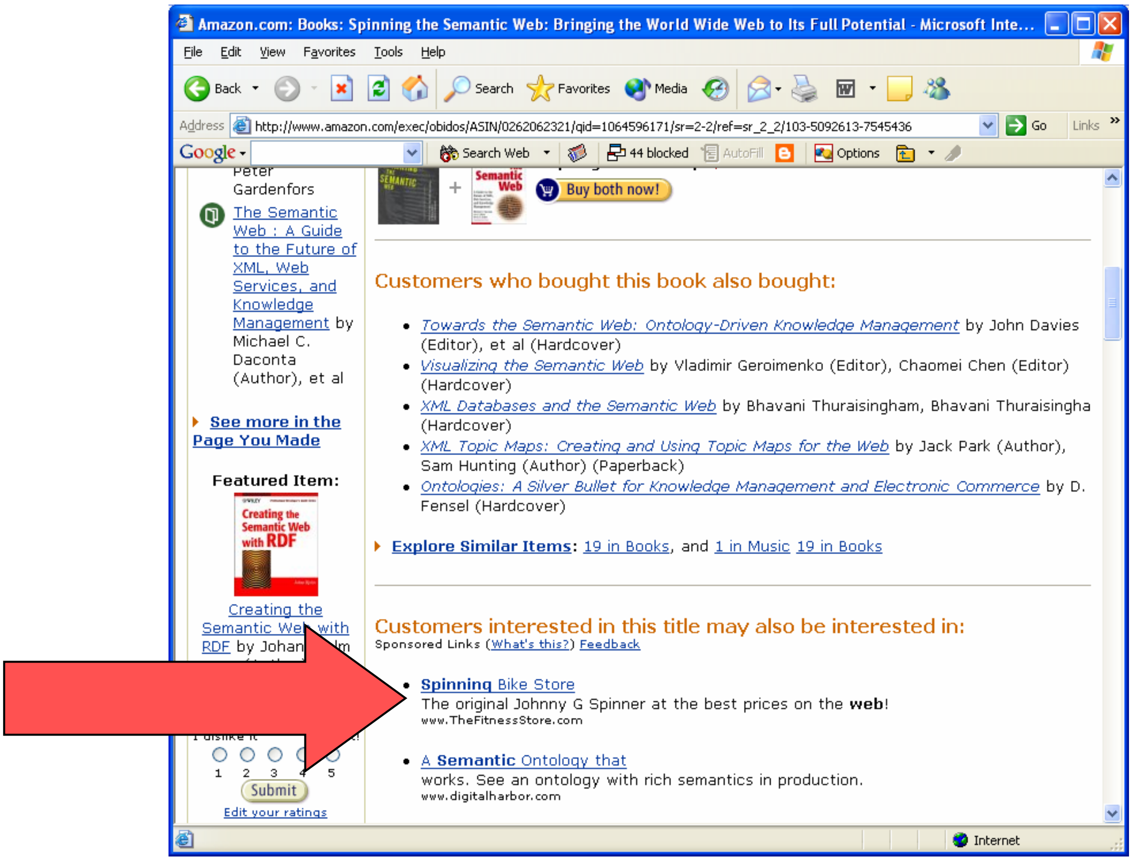
Edition: Hardcover

See more product details

Better Together

Semantic Web / Motivation (2)




Amazon.com: Books: Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential - Microsoft Inte...
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://www.amazon.com/exec/obidos/ASIN/0262062321/qid=1064596171/sr=2-2/ref=sr_2_2/103-5092613-7545436
Go Links
Google Search Web 44 blocked AutoFill Options
Buy both now!
Featured Item:
Creating the Semantic Web with RDF
I dislike it
1 2 3 4 5
Submit
Edit your ratings
Internet

Peter Gardenfors

The Semantic Web : A Guide to the Future of XML, Web Services, and Knowledge Management by Michael C. Daconta (Author), et al

See more in the Page You Made

Creating the Semantic Web with RDF by Johan ...m

Customers who bought this book also bought:

- *Towards the Semantic Web: Ontology-Driven Knowledge Management* by John Davies (Editor), et al (Hardcover)
- *Visualizing the Semantic Web* by Vladimir Geroimenko (Editor), Chaomei Chen (Editor) (Hardcover)
- *XML Databases and the Semantic Web* by Bhavani Thuraisingham, Bhavani Thuraisingha (Hardcover)
- *XML Topic Maps: Creating and Using Topic Maps for the Web* by Jack Park (Author), Sam Hunting (Author) (Paperback)
- *Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce* by D. Fensel (Hardcover)

Explore Similar Items: 19 in Books, and 1 in Music 19 in Books

Customers interested in this title may also be interested in:
Sponsored Links (What's this?) Feedback

- **Spinning** Bike Store
  The original Johnny G Spinner at the best prices on the **web**!
  www.TheFitnessStore.com
- A **Semantic** Ontology that
  works. See an ontology with rich semantics in production.
  www.digitalharbor.com

Semantic Web / Motivation (3)

- Next generation web pages: Machine accessible semantics

- Search engines
  - concepts, not keywords
  - semantic narrowing/widening of queries
- Shopbots
  - semantic interchange, not screenscraping
- E-commerce
  - negotiation, catalogue mapping, personalization
- Web Services
  - need semantic characterizations to find them
- Navigation
  - by semantic proximity, not hardwired links

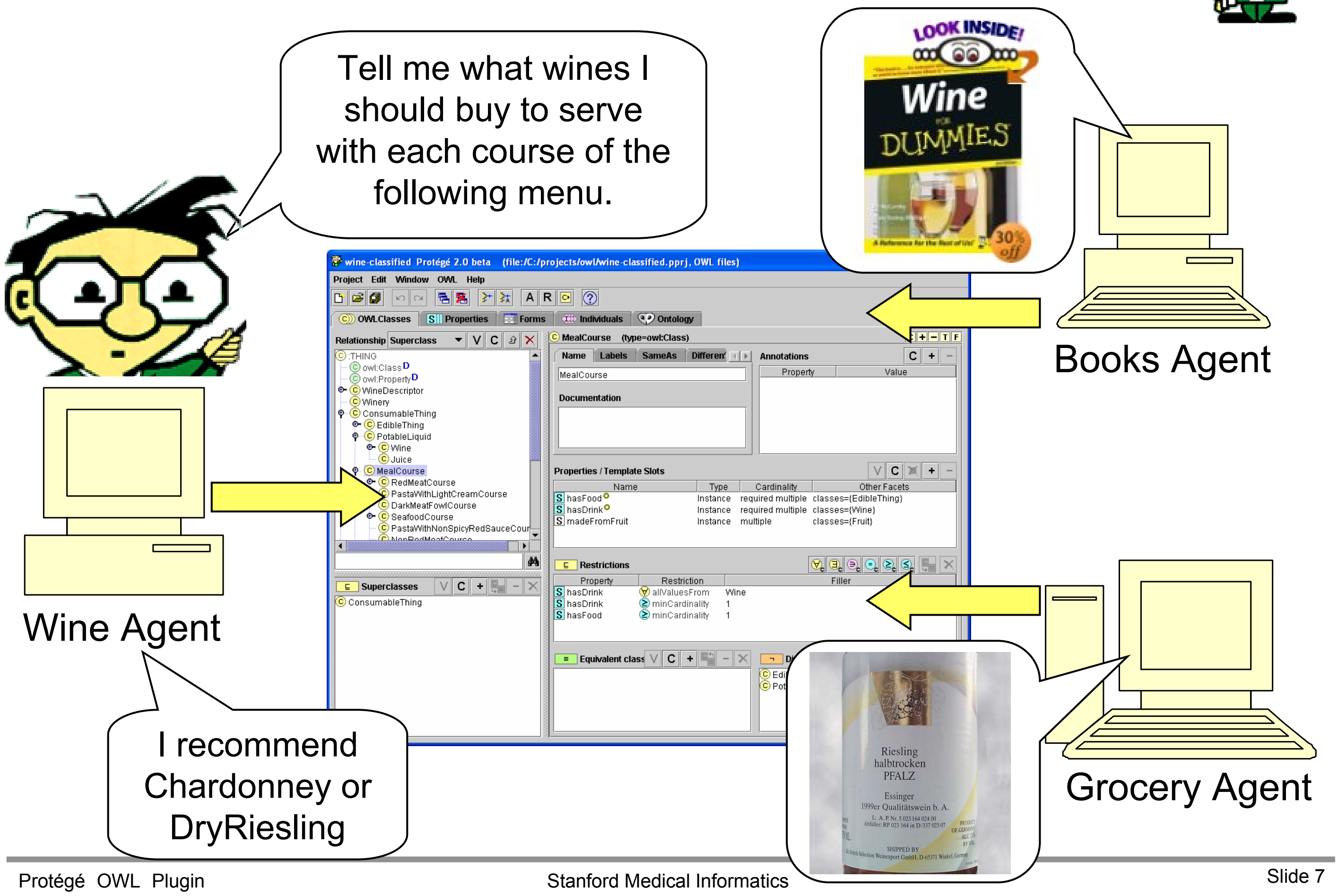
Semantic Web / Ontologies

- Formal, explicit specification of a shared conceptualization

- **What’s inside an ontology**
  - Classes + class-hierarchy
  - Properties (Slots) / values
  - Relations between classes (inheritance, disjoint, equivalent)
  - Restrictions on properties (type, cardinality)
  - Characteristics of slots (symmetric, transitive, …)
  - (possibly) Individuals

- Reasoning tasks: classification, subsumption

Semantic Web / Ontologies / Example Scenario



Tell me what wines I should buy to serve with each course of the following menu.

Wine Agent

I recommend Chardonney or DryRiesling

Books Agent

Grocery Agent

wine-classified Protégé 2.0 beta (file:/C:/projects/owl/wine-classified.pprj, OWL files)

Project Edit Window OWL Help

OWLClasses | Properties | Forms | Individuals | Ontology

Relationship: Superclass

- :THING
  - owl:Class
  - owl:Property
  - WineDescriptor
  - Winery
  - ConsumableThing
    - EdibleThing
    - PotableLiquid
      - Wine
      - Juice
    - MealCourse
      - RedMeatCourse
      - PastaWithLightCreamCourse
      - DarkMeatFowlCourse
      - SeafoodCourse
      - PastaWithNonSpicyRedSauceCour
      - NonRedMeatCourse

Superclasses: ConsumableThing

MealCourse (type=owl:Class)

Name: MealCourse

Documentation

Annotations

Property	Value

Properties / Template Slots

Name	Type	Cardinality	Other Facets
hasFood	Instance	required multiple	classes={EdibleThing}
hasDrink	Instance	required multiple	classes={Wine}
madeFromFruit	Instance	multiple	classes={Fruit}

Restrictions

Property	Restriction	Filler
hasDrink	allValuesFrom	Wine
hasDrink	minCardinality	1
hasFood	minCardinality	1

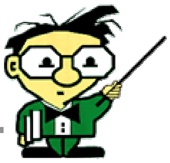
Equivalent class

Semantic Web / OWL

- Web Ontology Language (OWL)
- Developed by a World Wide Web Consortium (W3C) working group
- Based on DAML+OIL

- Semantic Web Vision: To enable machines to **comprehend** semantic documents and data
- Habitat for Autonomous Agents
- OWL facilitates greater machine readability of Web content than XML
- Extends RDF and RDF Schema by providing additional vocabulary along with a formal semantics

Semantic Web / OWL / Language Overview

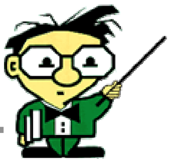


- Classes
- Properties
  - DatatypeProperties (boolean, float, integer, string)
  - ObjectProperties (relationships between classes)
- Individuals

- Built-in ontology mapping support (equivalent classes, sameAs)
- Some other property types (e.g., symmetric, transitive, functional)

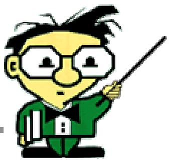
- Class Descriptions
  - can be used instead of named classes (e.g., to define superclasses)
  - define classes by the attributes of their members
    - enumerations red, green, or blue
    - restrictions all individuals that have **at least** 2 children
    - logical statements Person **and not** Student and **not** blue eyes

Protégé



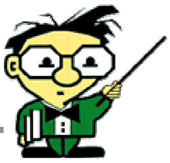
- An extensible and customizable toolset for constructing ontologies and for developing applications that use these ontologies
- Outstanding features
  - Automatic generation of graphical-user interfaces, based on user-defined models, for acquiring domain instances
  - Extensible knowledge model and architecture
  - Possible embedding of standalone applications in Protégé knowledge engineering environment and vice versa
  - Scalability to very large knowledge bases

Protégé / Historical background: early days



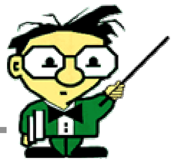
- ONCOCIN (1980s)
  - Clinical decision-support system (CDSS) for management of patients enrolled in cancer clinical trials
- OPAL (~1985)
  - A graphical user interface to encode cancer clinical trials for ONCOCIN based on a model of cancer trials
- Protégé (Mark Musen dissertation)
  - A system to define model of trials for any domain, to generate OPAL for eONCOCIN (CDSS for any trial domain)

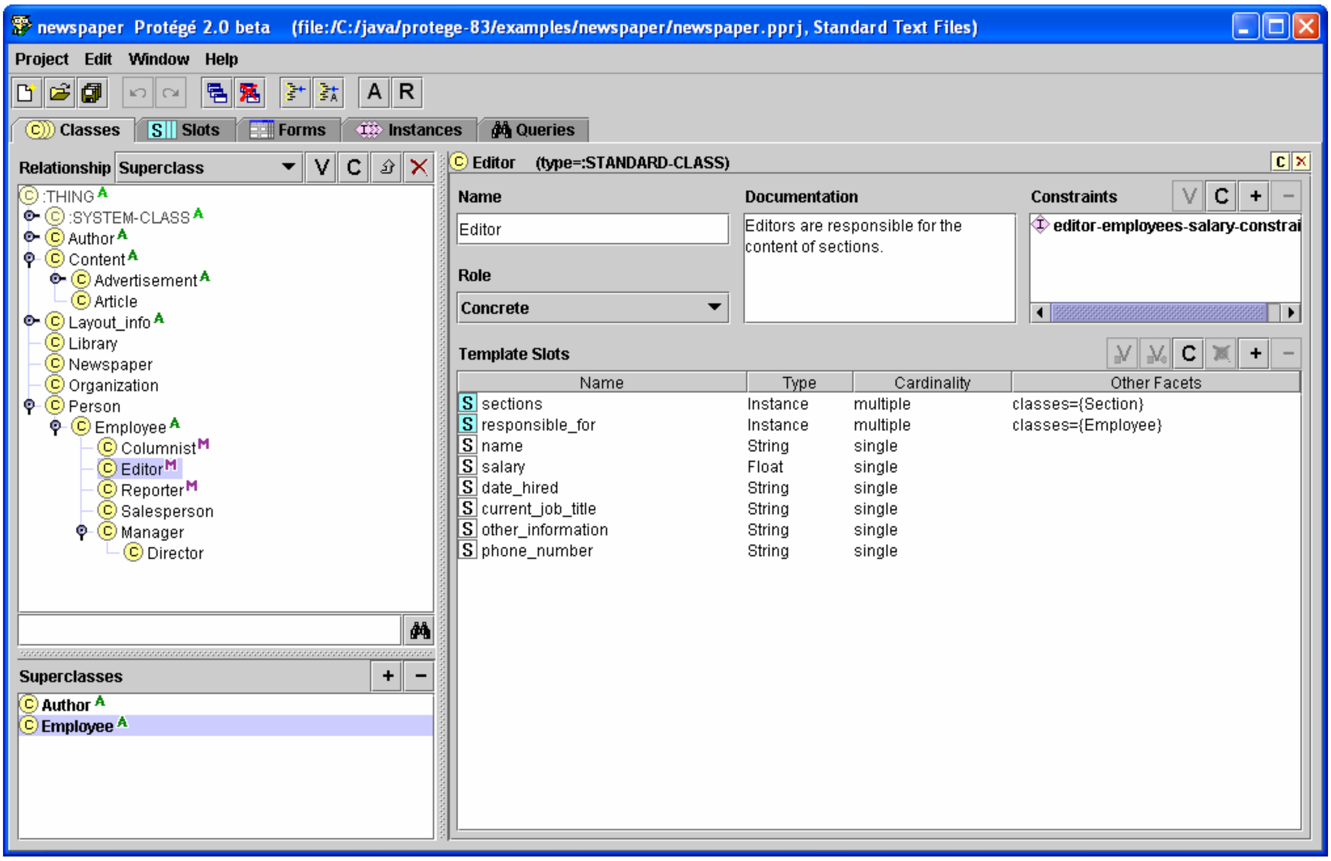
Protégé / Historical background: 1990s – present



- Protégé-II (early 1990s)
  - A knowledge engineering environment (on NeXTStep platform) to define model and generate GUI editor for any domain
- ProtégéWin (mid 1990s)
  - Windows version that emphasized usability
  - External user groups
- Protégé-2000 (late 1990 – present)
  - Java-based version that emphasized formal knowledge model and interoperability
  - Development of extensible plugin architecture
  - Open source
  - Renamed to Protégé in Version 2.0 (Fall, 2003)
  - OWL Support (work in progress, reasonably stable since Summer 2003)

Protégé / General Concepts / Classes





newspaper Protégé 2.0 beta (file:/C:/java/protege-83/examples/newspaper/newspaper.pprj, Standard Text Files)

Relationship: Superclass

- :THING
  - :SYSTEM-CLASS
  - Author
  - Content
    - Advertisement
    - Article
  - Layout_info
  - Library
  - Newspaper
  - Organization
  - Person
    - Employee
      - Columnist
      - Editor
      - Reporter
      - Salesperson
      - Manager
        - Director

Superclasses
- Author
- Employee

Editor (type=:STANDARD-CLASS)

Name: Editor

Documentation: Editors are responsible for the content of sections.

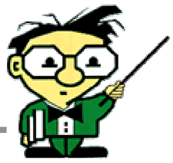
Constraints: editor-employees-salary-constrai

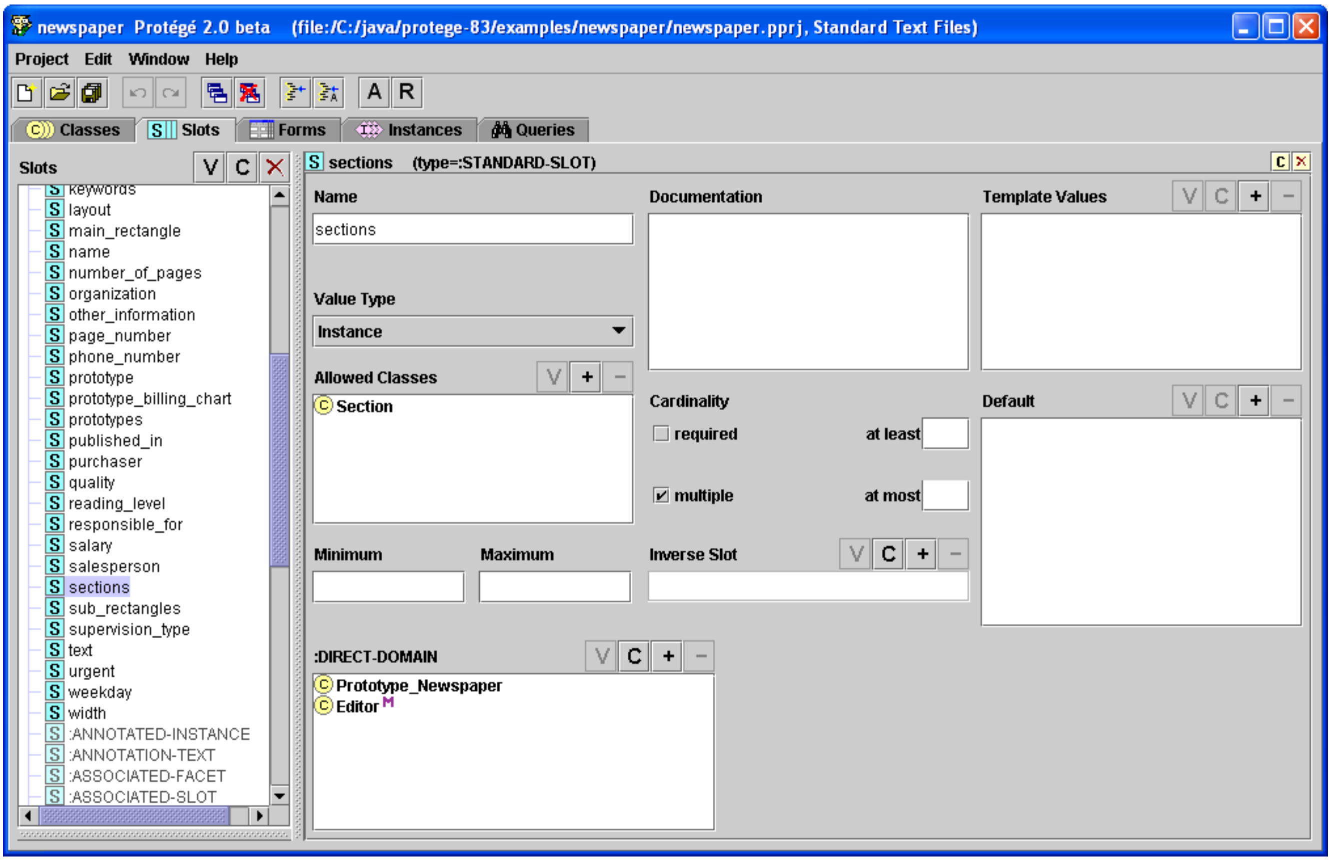
Role: Concrete

Template Slots

Name	Type	Cardinality	Other Facets
sections	Instance	multiple	classes={Section}
responsible_for	Instance	multiple	classes={Employee}
name	String	single	
salary	Float	single	
date_hired	String	single	
current_job_title	String	single	
other_information	String	single	
phone_number	String	single	

Protégé / General Concepts / Slots (Properties)





newspaper Protégé 2.0 beta (file:/C:/java/protege-83/examples/newspaper/newspaper.pprj, Standard Text Files)

Slots
- keywords
- layout
- main_rectangle
- name
- number_of_pages
- organization
- other_information
- page_number
- phone_number
- prototype
- prototype_billing_chart
- prototypes
- published_in
- purchaser
- quality
- reading_level
- responsible_for
- salary
- salesperson
- sections
- sub_rectangles
- supervision_type
- text
- urgent
- weekday
- width
- :ANNOTATED-INSTANCE
- :ANNOTATION-TEXT
- :ASSOCIATED-FACET
- :ASSOCIATED-SLOT

sections (type=:STANDARD-SLOT)

Name: sections

Documentation:

Template Values:

Value Type: Instance

Allowed Classes: Section

Cardinality
- □ required at least
- ☑ multiple at most

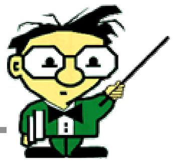
Default:

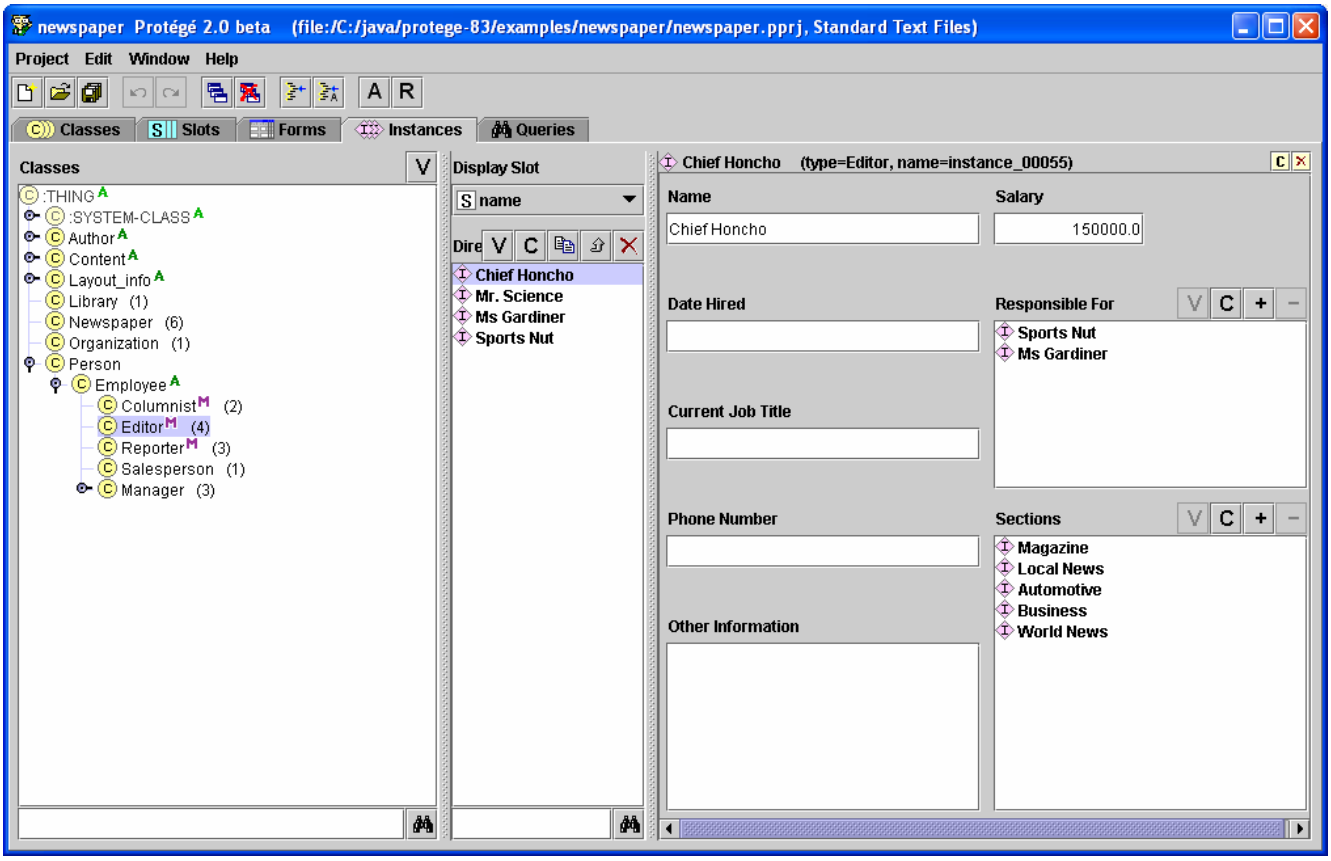
Minimum: Maximum:

Inverse Slot:

:DIRECT-DOMAIN
- Prototype_Newspaper
- Editor

Protégé / General Concepts / Instances





newspaper Protégé 2.0 beta (file:/C:/java/protege-83/examples/newspaper/newspaper.pprj, Standard Text Files)
Project Edit Window Help
A R
Classes
Slots
Forms
Instances
Queries

Classes

- :THING
  - :SYSTEM-CLASS
  - Author
  - Content
  - Layout_info
  - Library (1)
  - Newspaper (6)
  - Organization (1)
  - Person
    - Employee
      - Columnist (2)
      - Editor (4)
      - Reporter (3)
      - Salesperson (1)
      - Manager (3)

Display Slot: name

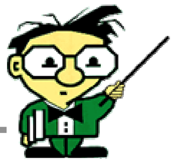
Direct Instances

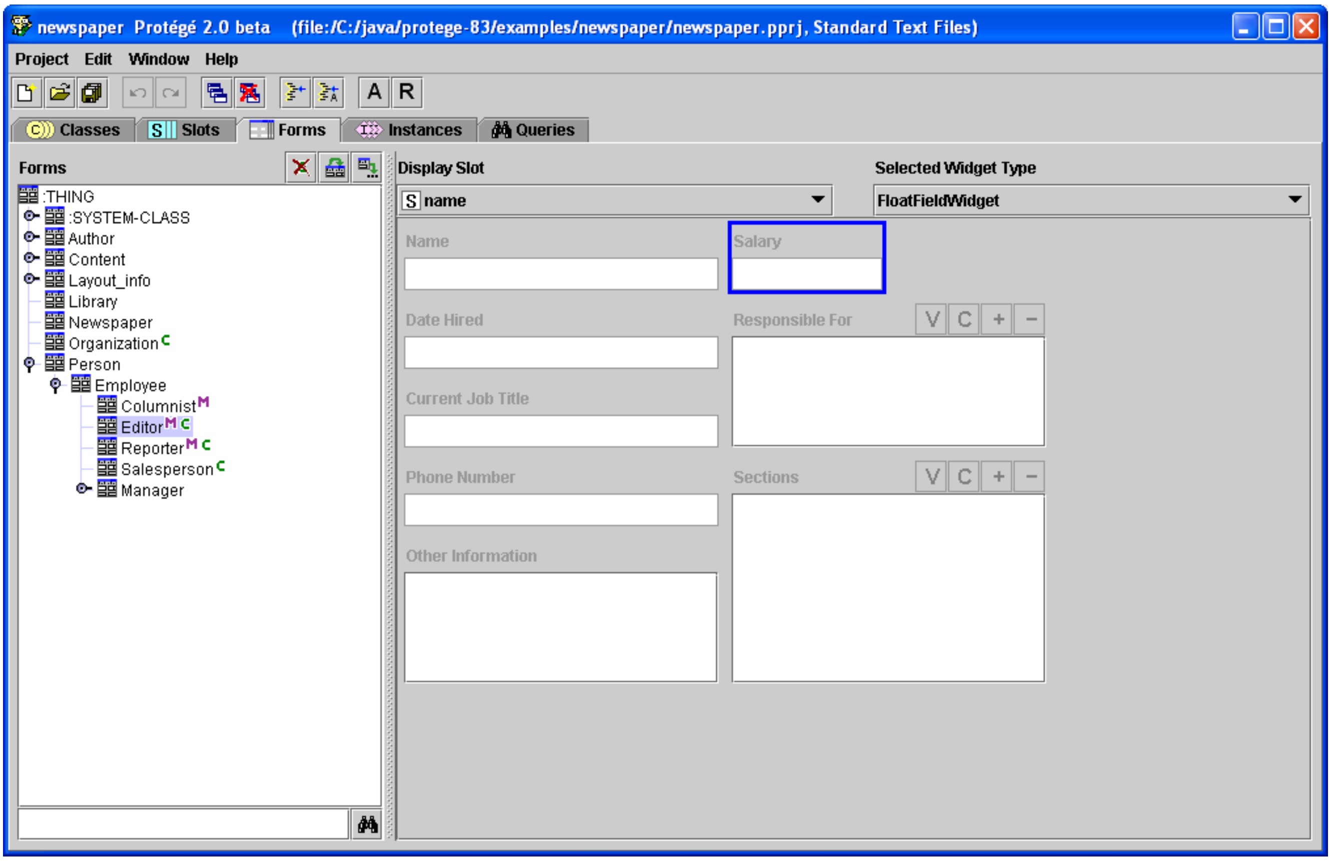
- Chief Honcho
- Mr. Science
- Ms Gardiner
- Sports Nut

Chief Honcho (type=Editor, name=instance_00055)

Name: Chief Honcho	Salary: 150000.0
Date Hired:	Responsible For: Sports Nut Ms Gardiner
Current Job Title:	
Phone Number:	Sections: Magazine Local News Automotive Business World News
Other Information:	

Protégé / General Concepts / Forms





newspaper Protégé 2.0 beta (file:/C:/java/protege-83/examples/newspaper/newspaper.pprj, Standard Text Files)
Project Edit Window Help
A R
Classes
Slots
Forms
Instances
Queries

Forms

- :THING
  - :SYSTEM-CLASS
  - Author
  - Content
  - Layout_info
  - Library
  - Newspaper
  - Organization
  - Person
    - Employee
      - Columnist
      - Editor
      - Reporter
      - Salesperson
      - Manager

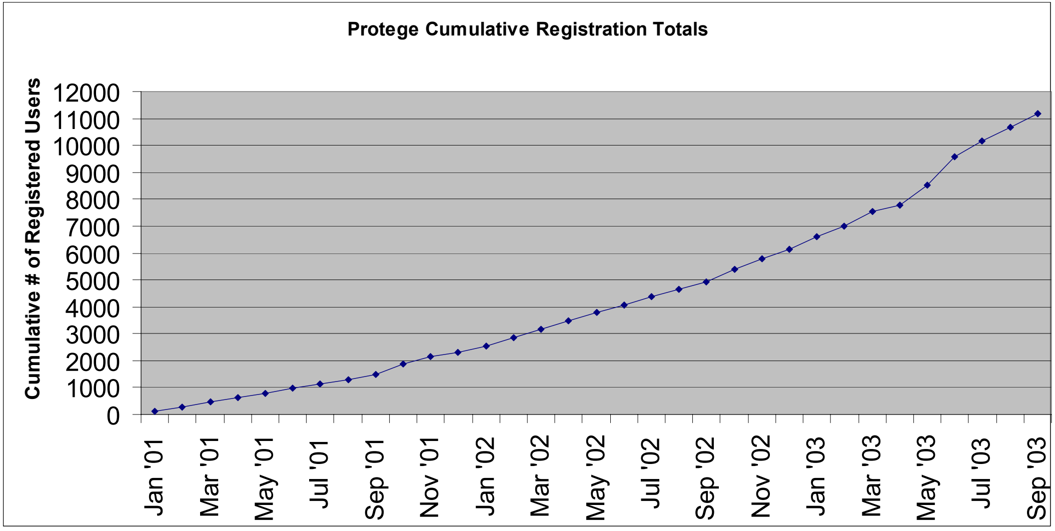
Display Slot: name

Selected Widget Type: FloatFieldWidget

Name	Salary
Date Hired	Responsible For
Current Job Title	
Phone Number	Sections
Other Information	

Protégé / A world-wide user community

- Stanford offers support to individual users via “protege-help” mailing list
- Users support one another and brainstorm about new ideas using the “protege-discussion” mailing list
- Protégé Web pages provide access to contributed plug-ins, ontologies, help manuals, FAQs, and scientific publications
- Every year, we hold a Protégé users group meeting for both technical discussions and schmoozing



Protégé / Lots of user-contributed “plug ins”

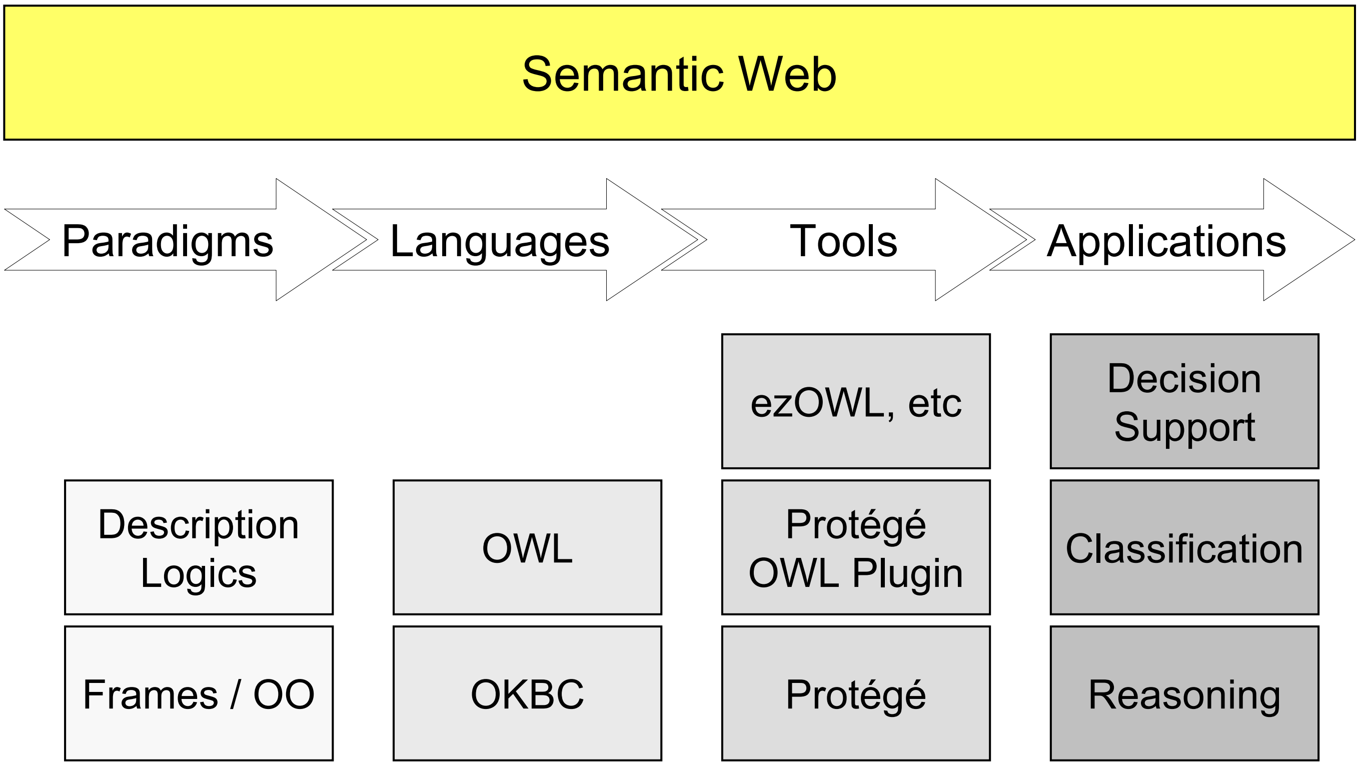
- Like Web browsers, Protégé-2000 accepts a wide range of “plug ins” that enhance its functionality
- Many of these plug ins are contributed by members of our user community with little or no discussion with our development team
- It’s absolutely amazing to see what our user community has contributed back to us!

- New visualization systems
- New inferencing systems
- New import and export formats
- New user-interface features
- New means of accessing external data sources

Protégé / Storage Formats

- Users edit and view ontologies in a manner that insulates them from the ultimate storage format
- Ontologies may be read in from, written out to, and interconverted between a large number of formats
  - Relational databases (ODBC)
  - CLIPS
  - UML / XMI
  - XML / XML Schema
  - RDF
  - Topic Maps
  - DAML+OIL
  - OWL

Protégé / Role of Protégé in the Semantic Web



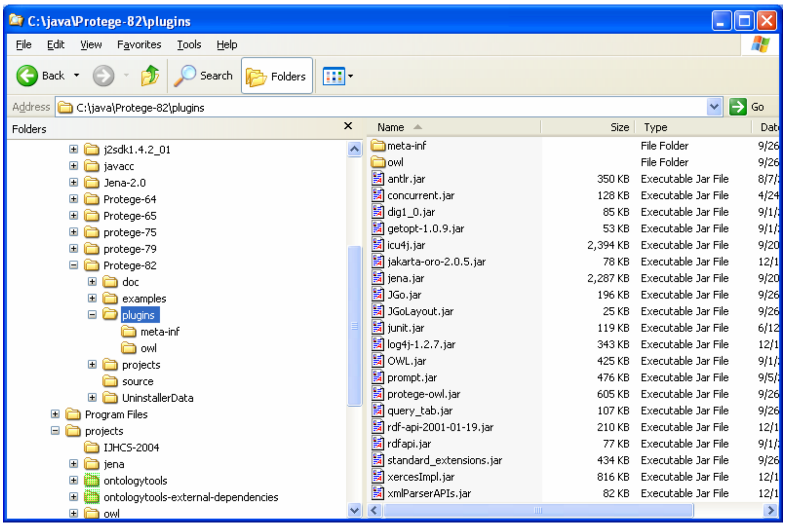
Protégé / OWL Plugin

- Extension of Protégé to allow editing OWL ontologies
- Project started April 2003, based on ideas from previous projects (OilTab, RDF and DAML+OIL backends)
- Currently in beta release (http://protege.stanford.edu/plugins/owl)

- Features
  - Loading and saving OWL files
  - Graphical editors for class expressions
  - Access to description logics inference components such as classifiers
  - Powerful platform for hooking in custom-tailored components

Protégé / OWL / Installation

1. Install Java 2 Virtual Machine (SDK version 1.4.2)
2. Install latest version of Protégé 2.0
   http://protege.stanford.edu/download/prerelease
3. Download OWL Plugin and unzip it into plugins folder
   http://protege.stanford.edu/plugins/owl/download/protege-owl.zip
   (This will create a folder plugins/owl containing two files)



C:\java\Protege-82\plugins

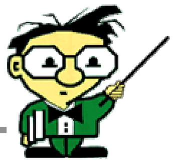
```
j2sdk1.4.2_01
javacc
Jena-2.0
Protege-64
Protege-65
protege-75
protege-79
Protege-82
  doc
  examples
  plugins
    meta-inf
    owl
  projects
  source
  UninstallerData
Program Files
projects
  IJHCS-2004
  jena
  ontologytools
  ontologytools-external-dependencies
  owl
```

Name	Size	Type	Dat
meta-inf		File Folder	9/26
owl		File Folder	9/26
antlr.jar	350 KB	Executable Jar File	8/7/
concurrent.jar	128 KB	Executable Jar File	4/24
dig1_0.jar	85 KB	Executable Jar File	9/1/
getopt-1.0.9.jar	53 KB	Executable Jar File	9/1/
icu4j.jar	2,394 KB	Executable Jar File	9/20
jakarta-oro-2.0.5.jar	78 KB	Executable Jar File	12/1
jena.jar	2,287 KB	Executable Jar File	9/20
JGo.jar	196 KB	Executable Jar File	9/26
JGoLayout.jar	25 KB	Executable Jar File	9/26
junit.jar	119 KB	Executable Jar File	6/12
log4j-1.2.7.jar	343 KB	Executable Jar File	12/1
OWL.jar	425 KB	Executable Jar File	9/1/
prompt.jar	476 KB	Executable Jar File	9/5/
protege-owl.jar	605 KB	Executable Jar File	9/26
query_tab.jar	107 KB	Executable Jar File	9/26
rdf-api-2001-01-19.jar	210 KB	Executable Jar File	12/1
rdfapi.jar	77 KB	Executable Jar File	9/1/
standard_extensions.jar	434 KB	Executable Jar File	9/26
xercesImpl.jar	816 KB	Executable Jar File	12/1
xmlParserAPIs.jar	82 KB	Executable Jar File	12/1

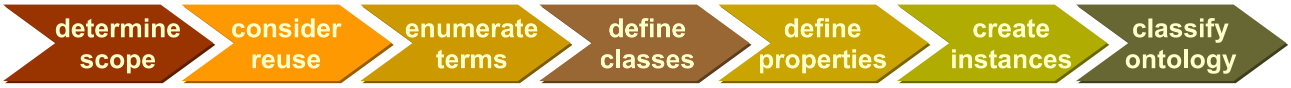
Optional Components

- Install other plugins such as ezOWL
- Install classifiers such as Racer

Protégé / OWL / Ontology Development Process

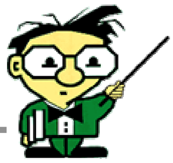


- Ontology development is an iterative process
- Roughly consists of the following activities

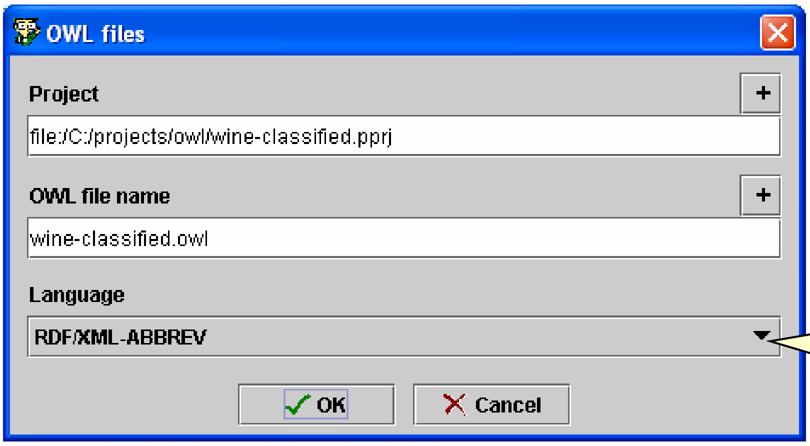


- Ontology development requires
  - some experience and foresight
  - communication between domain experts and developers
  - a tool that is easy to understand, yet powerful
  - a tool that supports ontology evolution

Protégé / OWL / Files and Projects



- An OWL Project typically consist of two files
  - an .owl file (e.g. wine-classified.owl): Contains the ontology itself as RDF
  - a .pprj file (e.g. wine-classified.pprj): Contains project metadata such as layout information, file names
- Project/New... OWL Files – Creates a new (empty) project
- Project/Open... – Loads an existing project
- Project/Save as... – Saves the current project under a given name



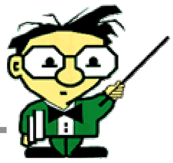
Project: file:/C:/projects/owl/wine-classified.pprj

OWL file name: wine-classified.owl

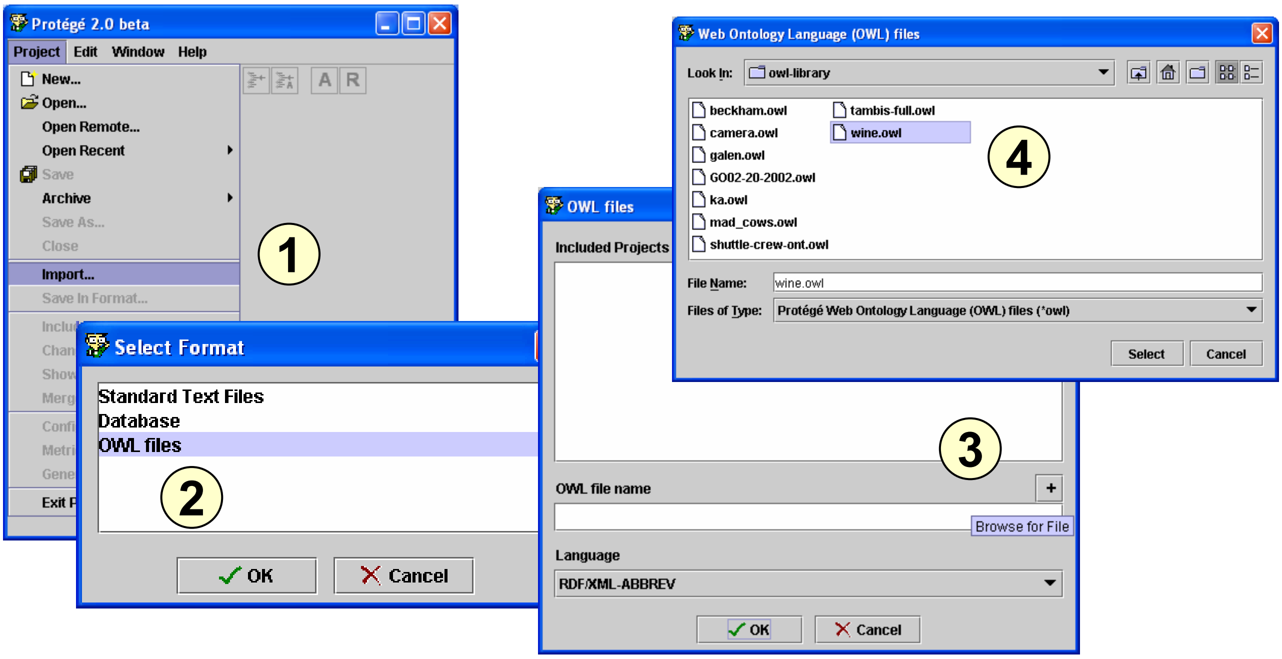
Language: RDF/XML-ABBREV

Alternative serializations of OWL (e.g. N3)

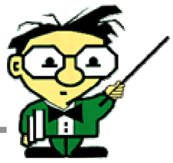
Protégé / OWL / Files and Projects / Import

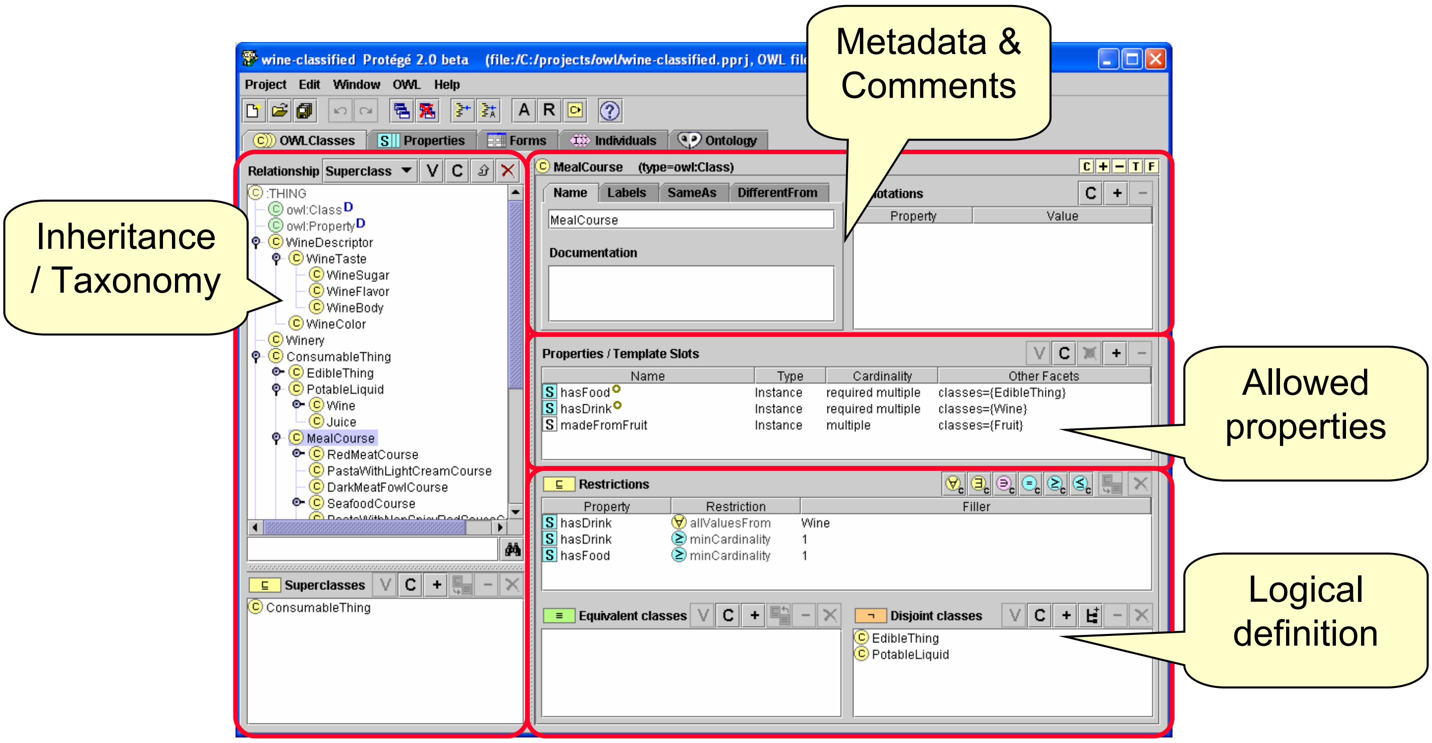


- Use Project/Import... to load an existing .owl file that was
  - developed with another tool
  - developed with a previous version of Protégé (.pprj file might be outdated)



Protégé / OWL / Classes / Tab





Name	Type	Cardinality	Other Facets
hasFood	Instance	required multiple	classes={EdibleThing}
hasDrink	Instance	required multiple	classes={Wine}
madeFromFruit	Instance	multiple	classes={Fruit}

Property	Restriction	Filler
hasDrink	allValuesFrom	Wine
hasDrink	minCardinality	1
hasFood	minCardinality	1

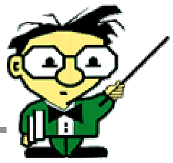
Metadata & Comments

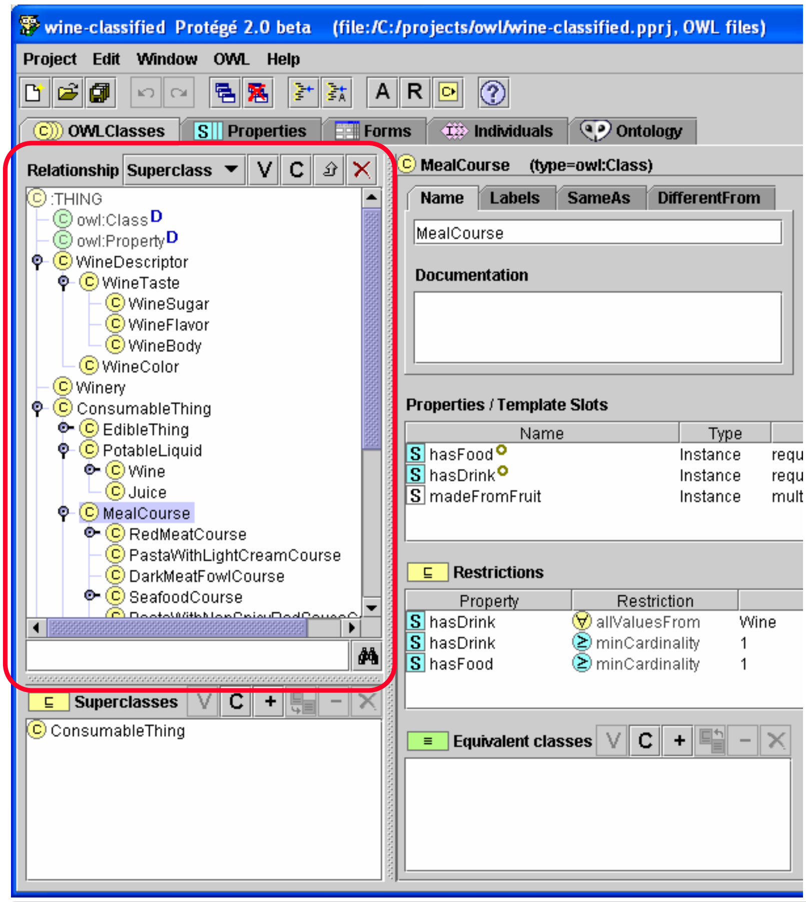
Inheritance / Taxonomy

Allowed properties

Logical definition

Protégé / OWL / Classes / Inheritance Tree (1)





wine-classified Protégé 2.0 beta (file:/C:/projects/owl/wine-classified.pprj, OWL files)

Name: MealCourse

Documentation

Properties / Template Slots

Name	Type	requ
S hasFood	Instance	requ
S hasDrink	Instance	requ
S madeFromFruit	Instance	mult

Restrictions

Property	Restriction	
S hasDrink	allValuesFrom	Wine
S hasDrink	minCardinality	1
S hasFood	minCardinality	1

Equivalent classes

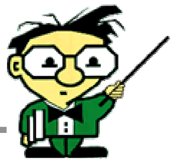
- Displays subclass / superclass relationship
- :THING is top-level class (owl:Thing in OWL)

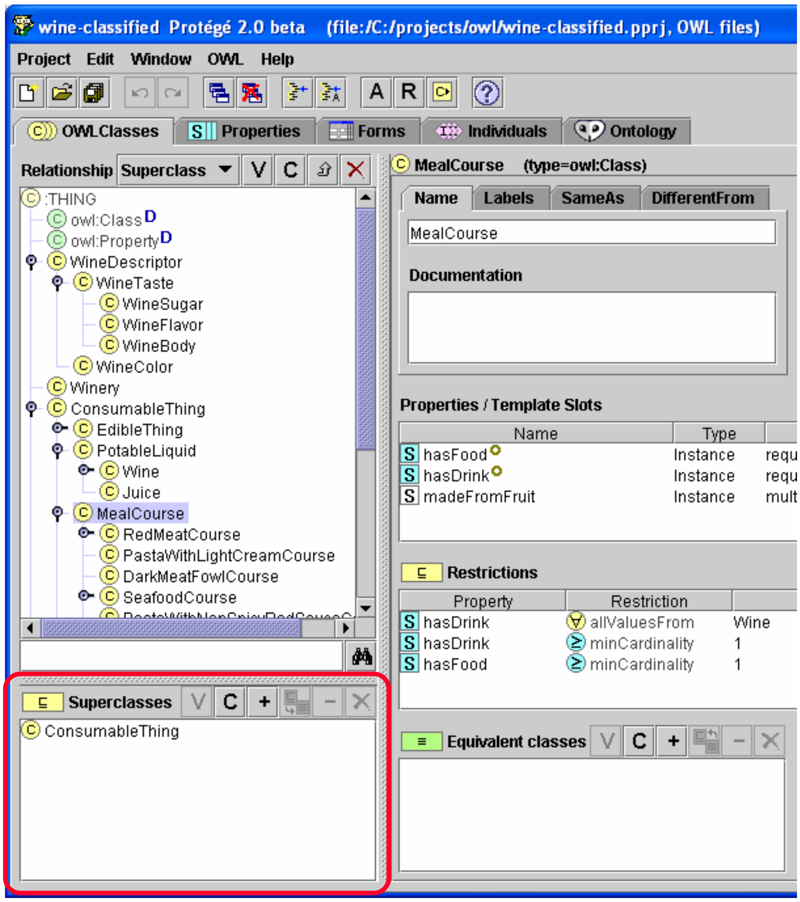
V View / Edit class in extra window

C Create subclass

✗ Delete class

Protégé / OWL / Classes / Inheritance Tree (2)





wine-classified Protégé 2.0 beta (file:/C:/projects/owl/wine-classified.pprj, OWL files)

Name: MealCourse

Documentation

Properties / Template Slots

Name	Type	requ
S hasFood	Instance	requ
S hasDrink	Instance	requ
S madeFromFruit	Instance	mult

Restrictions

Property	Restriction	
S hasDrink	allValuesFrom	Wine
S hasDrink	minCardinality	1
S hasFood	minCardinality	1

Equivalent classes

- Drag and drop
- Multiple inheritance
- More options in popup menu

V View / Edit superclass

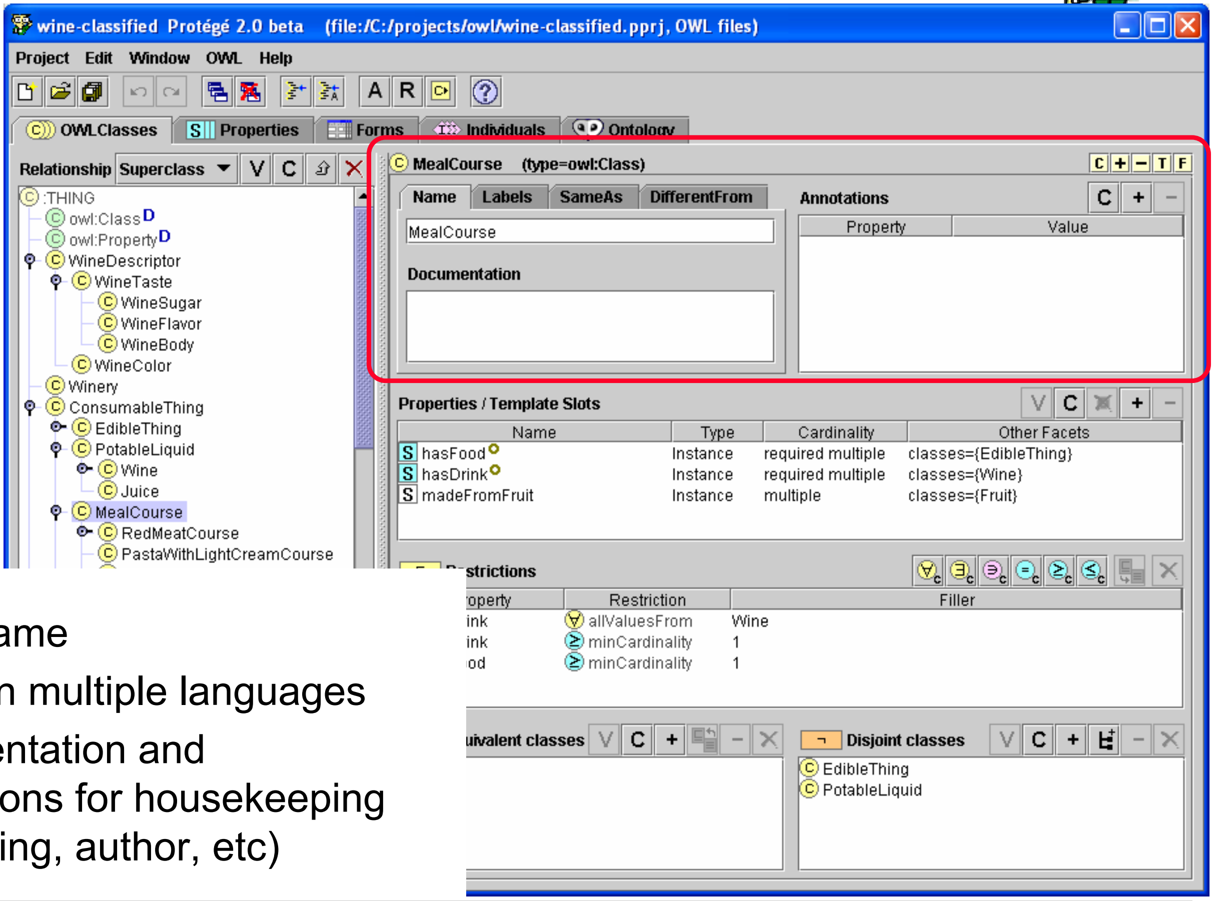
C Create superclass

+ Add superclass

– Remove superclass

✗ Delete class expression

Protégé / OWL / Classes / Class metadata



wine-classified Protégé 2.0 beta (file:/C:/projects/owl/wine-classified.pprj, OWL files)

MealCourse (type=owl:Class)

Name: MealCourse

Documentation

Property	Value

Properties / Template Slots

Name	Type	Cardinality	Other Facets
hasFood	Instance	required multiple	classes={EdibleThing}
hasDrink	Instance	required multiple	classes={Wine}
madeFromFruit	Instance	multiple	classes={Fruit}

operty	Restriction	Filler
ink	allValuesFrom	Wine
ink	minCardinality	1
od	minCardinality	1

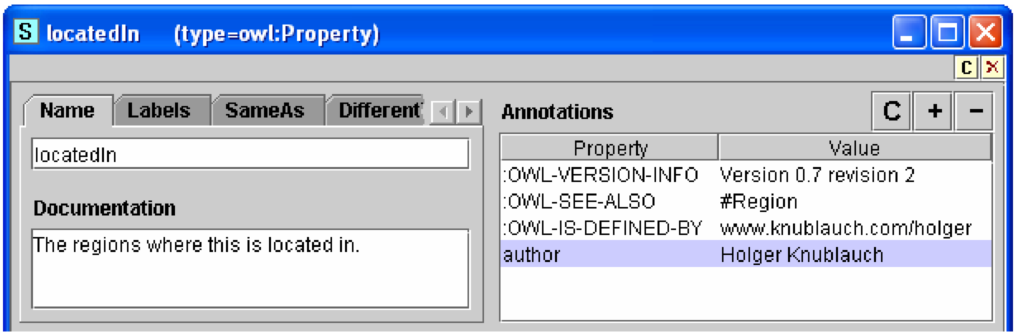
Disjoint classes: EdibleThing, PotableLiquid

- Class name
- Labels in multiple languages
- Documentation and annotations for housekeeping (versioning, author, etc)

Protégé / OWL / Resource Metadata

Standard metadata for all ontology resources

	OWL Property	Description
Name	rdf:ID	A unique name for the resource
Documentation	rdfs:comment	A comment describing the resource
Labels	rdfs:label	Labels in multiple languages (alternative to Name)
SameAs	owl:sameAs	List of resources that are known to be identical
DifferentFrom	owl:differentFrom	List of resources that are known to be not identical



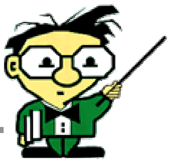
locatedIn (type=owl:Property)

Name: locatedIn

Documentation: The regions where this is located in.

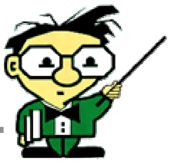
Property	Value
:OWL-VERSION-INFO	Version 0.7 revision 2
:OWL-SEE-ALSO	#Region
:OWL-IS-DEFINED-BY	www.knublauch.com/holger
author	Holger Knublauch

Protégé / OWL / Resource Metadata / Names

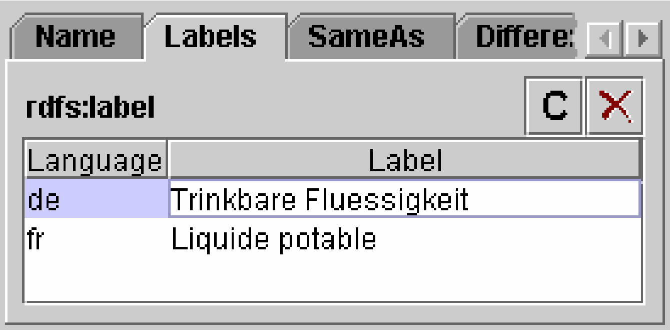


- Each resource (Class, Property or Individual) must have a unique name (within its namespace)
- Names must be the local part of a URI
- Support for multiple namespaces is in progress, and the editor will reflect this (e.g. displaying namespace placeholders such as wine:Riesling if multiple namespaces are used)
- In Protégé, names
  - must start with a letter or the underscore (_)
  - must only contain letters, digits, _, -, .
  - can not contain spaces
- Protégé will display illegal names in red and replace illegal characters with underscores (_)

Protégé / OWL / Resource Metadata / Labels



- Labels define alternative names of resources (classes, properties, individuals)
- Labels can be annotated with a language attribute (country key)
- International ontologies
- May be used for display purposes in later Protégé versions



Language	Label
de	Trinkbare Fluessigkeit
fr	Liquide potable

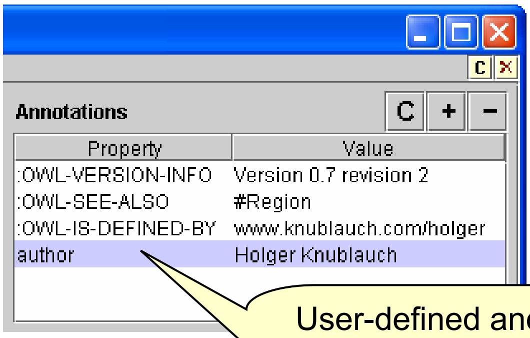
Protégé / OWL / Resource Metadata / Annotations

- Annotations are comments on the resource
- Have no formal semantics, are ignored during reasoning
- Must be values of properties that are marked as “Annotation Properties” (type owl:AnnotationProperty)
- Any annotation property can be assigned to any resource
- Some pre-defined annotation properties exist
- Currently only text strings are supported

C Create new ann. property

+ Add value for an existing ann. property

– Remove property value



Property	Value
:OWL-VERSION-INFO	Version 0.7 revision 2
:OWL-SEE-ALSO	#Region
:OWL-IS-DEFINED-BY	www.knublauch.com/holger
author	Holger Knublauch

User-defined and pre-defined annotation properties can be used

Protégé / OWL / Classes / Properties

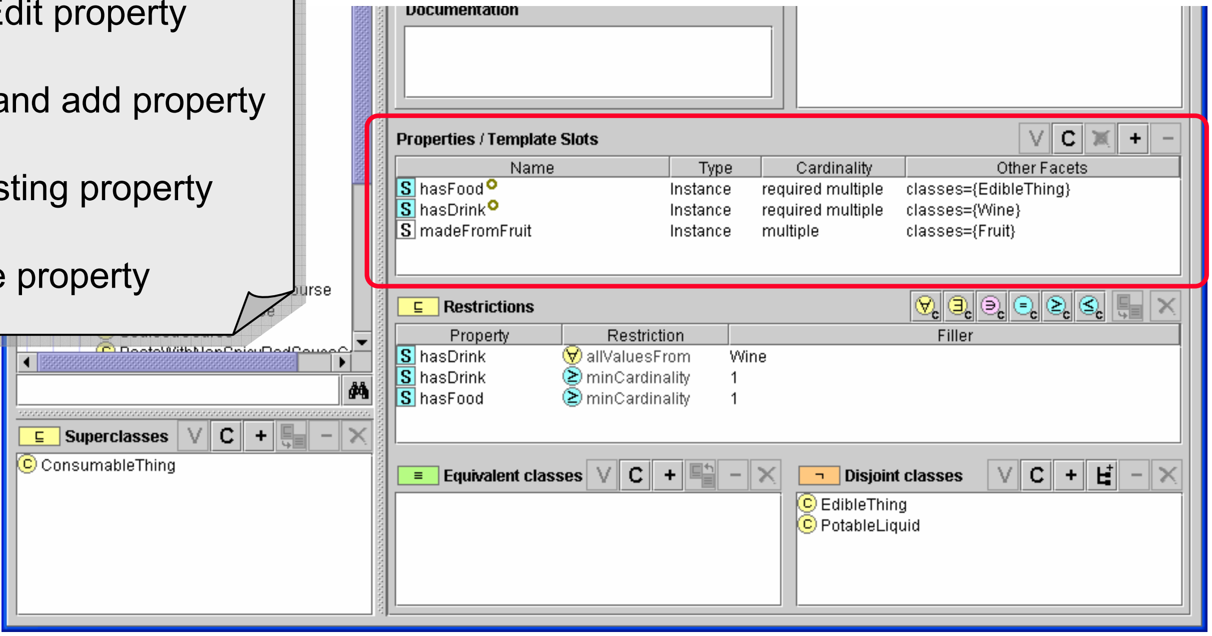
- List of all properties that can be assigned to individuals of this class (Either explicitly (property has the class as domain) or by use)
- Similar to attributes and relationships in object-oriented languages
- Inherited properties have a white S

V View / Edit property

C Create and add property

+ Add existing property

– Remove property



Name	Type	Cardinality	Other Facets
S hasFood	Instance	required multiple	classes={EdibleThing}
S hasDrink	Instance	required multiple	classes={Wine}
S madeFromFruit	Instance	multiple	classes={Fruit}

Property	Restriction	Filler
S hasDrink	∀ allValuesFrom	Wine
S hasDrink	≥ minCardinality	1
S hasFood	≥ minCardinality	1

Superclasses: ConsumableThing

Disjoint classes: EdibleThing, PotableLiquid

Protégé / OWL / Properties

- A property can be used to assign values to individuals (and classes and properties) and to define relations between them

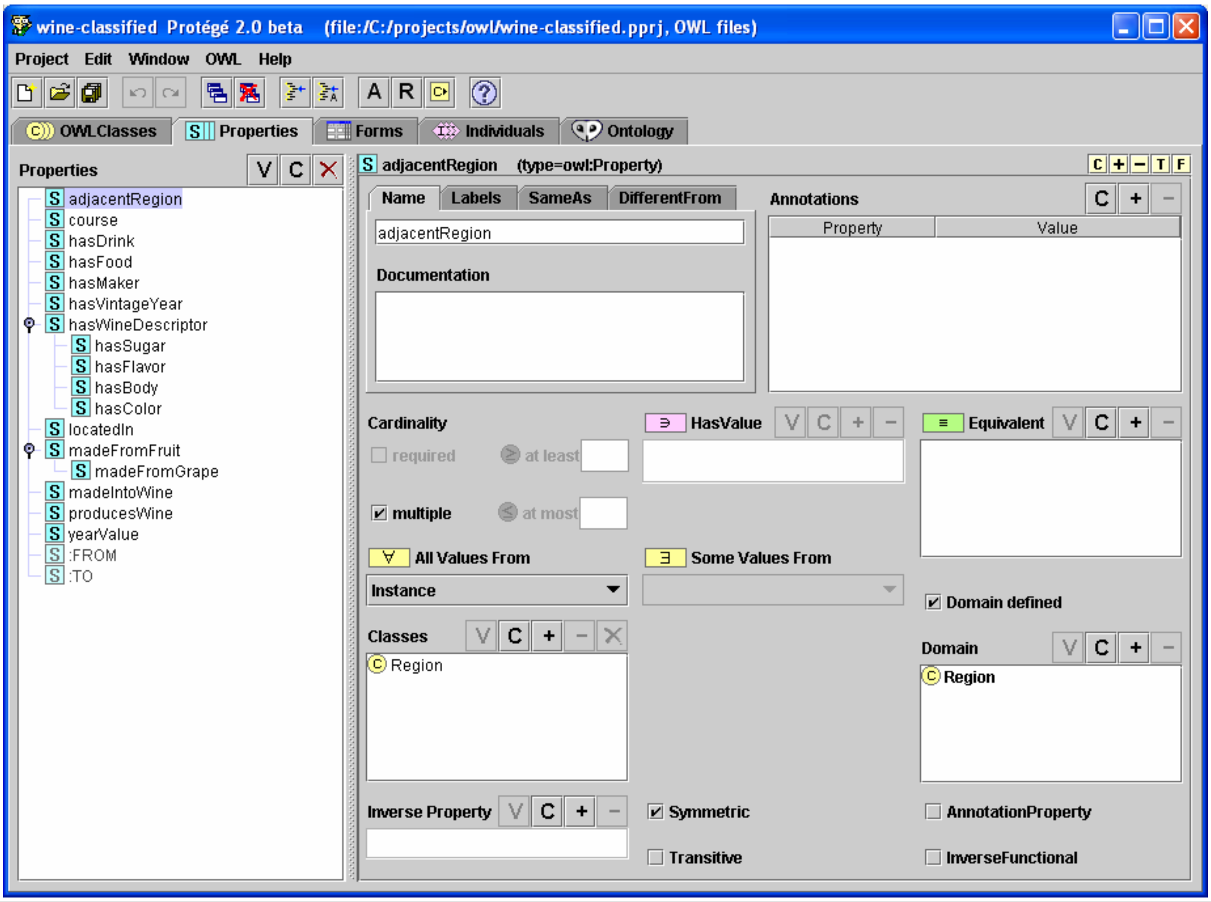
- Properties can have characteristics / attributes [These are defined in the Property metaclass]

- Properties have “global” and “local” characteristics
- Global characteristics define general properties of the property
- Local characteristics define properties when the property is assigned to a certain class
- Local attributes are represented by means of OWL restrictions (below)
- Some characteristics can have both global and local values (i.e. they are “overloaded” for a local class):
  - Range (global) = All Values From (local)
  - FunctionalProperty (global) = Maximum cardinality of 1 (local)

Protégé / OWL / Properties / Global Characteristics

	OWL Property	Description
Range	rdfs:range	The allowed datatype or classes for values
Domain	rdfs:domain	The classes where this can be assigned a value to
Domain Defined	(explicit domain)	Indicates whether there is an explicit domain
Super-properties	rdfs:subPropertyOf	A collection of the parent properties (inheritance)
Equivalents	owl:equivalent-Property	A collection of equivalent properties
Inverse Property	owl:inverseOf	The inverse property (bi-directional relationship)
Transitive	owl:Transitive-Property	If A and B are related, and B and C are related, then it follows that A and C are also related
Symmetric	owl:Symmetric-Property	If A is related to B, then B is also related to A
InverseFunctional	owl:Inverse-FunctionalProperty	Whether this is inverse functional (comparable to key in databases)
Multiple Cardinality	owl:Functional-Property	Whether this is a functional property (can have only one value, maximum cardinality of 1)
Annotation-Property	owl:Annotation-Property	Annotation properties are used only as comments and metadata – they are ignored during reasoning

Protégé / OWL / Properties / Properties Tab



wine-classified Protégé 2.0 beta (file:/C:/projects/owl/wine-classified.pprj, OWL files)

- adjacentRegion
- course
- hasDrink
- hasFood
- hasMaker
- hasVintageYear
- hasWineDescriptor
  - hasSugar
  - hasFlavor
  - hasBody
  - hasColor
- locatedIn
- madeFromFruit
  - madeFromGrape
- madeIntoWine
- producesWine
- yearValue
- :FROM
- :TO

adjacentRegion (type=owl:Property)

Name: adjacentRegion

Documentation

Cardinality

☐ required at least

☑ multiple at most

All Values From: Instance

Classes: Region

☑ Domain defined

Domain: Region

Inverse Property

☑ Symmetric

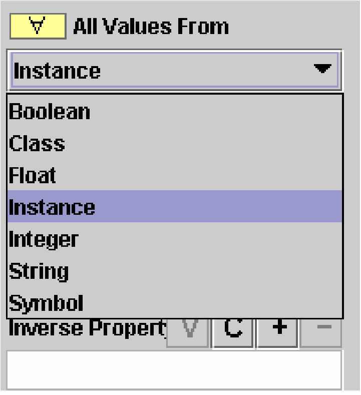
☐ Transitive

☐ AnnotationProperty

☐ InverseFunctional

Protégé / OWL / Properties / Range

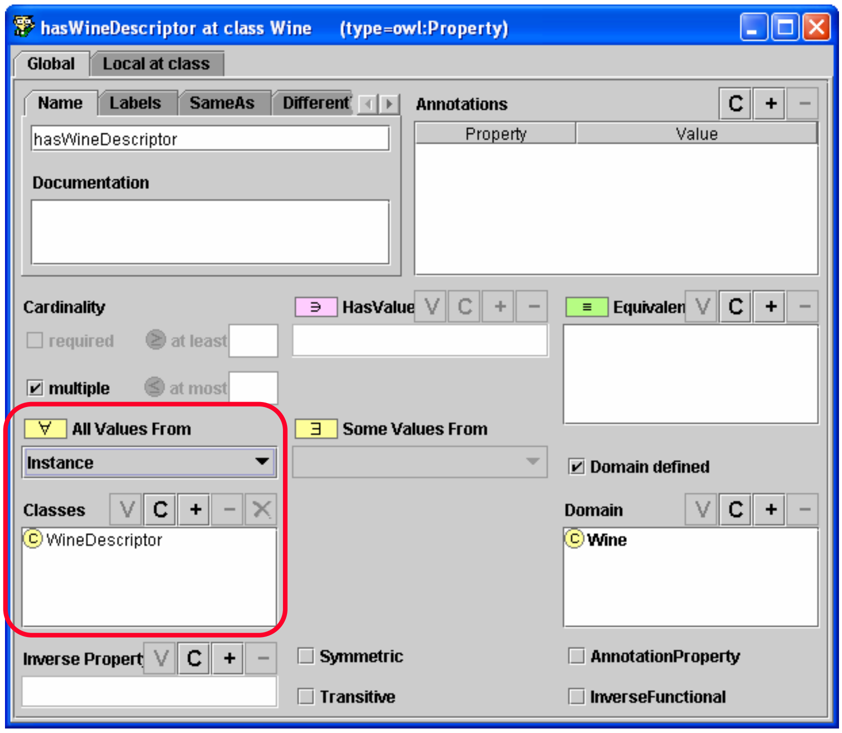
- Range defines the type of values that can be assigned to the property
- If the range is “Instance” then you also have to specify the list of allowed classes



All Values From

Instance

- Boolean
- Class
- Float
- Instance
- Integer
- String
- Symbol



hasWineDescriptor at class Wine (type=owl:Property)

Name: hasWineDescriptor

Documentation

Cardinality

☐ required at least

☑ multiple at most

All Values From: Instance

Classes: WineDescriptor

☑ Domain defined

Domain: Wine

☐ Symmetric

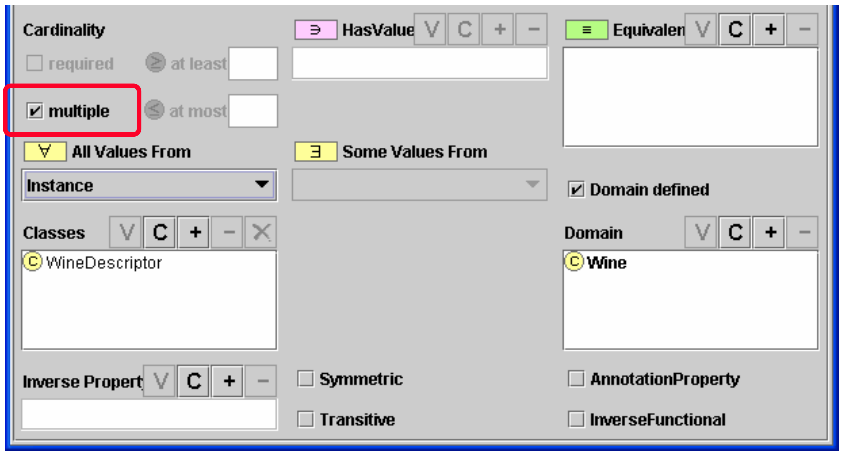
☐ Transitive

☐ AnnotationProperty

☐ InverseFunctional

Protégé / OWL / Properties / Global Cardinality

- The global cardinality states whether multiple values can be assigned to a given property
- Similar to 0..1 or 0..n relationships in relational and object-oriented models
- Properties with global cardinality of 0..1 in OWL are called functional properties
- Default cardinality is usually multiple



Cardinality

□ required   at least

☑ multiple   at most

HasValue

Equivalen

All Values From: Instance

Some Values From

☑ Domain defined

Classes: WineDescriptor

Domain: Wine

Inverse Propert

□ Symmetric

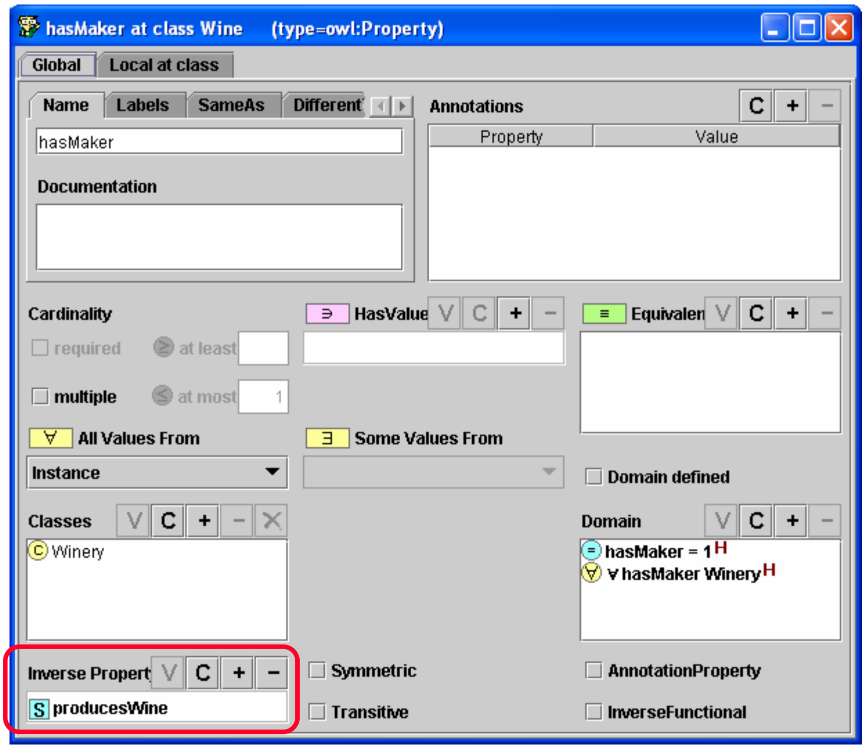
□ Transitive

□ AnnotationProperty

□ InverseFunctional

Protégé / OWL / Properties / Inverse Properties

- Inverse properties are bi-directional
- If Wine A **has maker** Winery B, then Winery B **produces** Wine A



hasMaker at class Wine (type=owl:Property)

Global | Local at class

Name | Labels | SameAs | Different

Name: hasMaker

Documentation

Annotations

Property	Value

Cardinality

□ required   at least

□ multiple   at most 1

HasValue

Equivalen

All Values From: Instance

Some Values From

□ Domain defined

Classes: Winery

Domain:

hasMaker = 1 H

∀ hasMaker Winery H

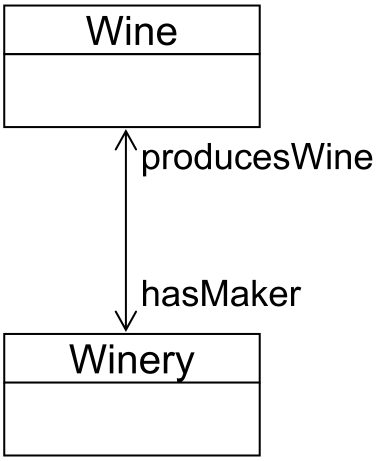
Inverse Propert: producesWine

□ Symmetric

□ Transitive

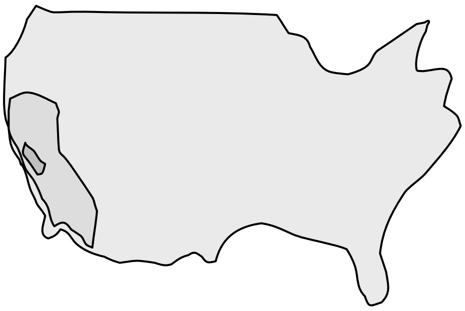
□ AnnotationProperty

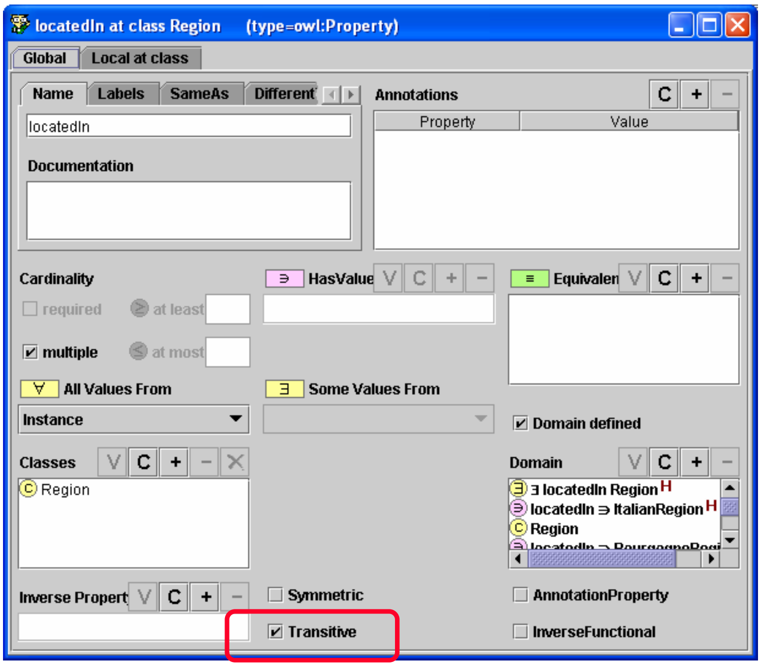
□ InverseFunctional



Protégé / OWL / Properties / Transitive

- Typically represent certain part-whole relations
- Example: Because the **SantaCruzMountainsRegion** is **locatedIn** the **CaliforniaRegion**, then it must also be locatedIn the **USRegion**, since **locatedIn** is transitive.





locatedIn at class Region (type=owl:Property)

Global | Local at class

Name | Labels | SameAs | Different

locatedIn

Documentation

Annotations

Property	Value

Cardinality

□ required ○ at least

☑ multiple ○ at most

∀ All Values From: Instance

∋ HasValue

∃ Some Values From

≡ Equivalent

☑ Domain defined

Classes: Ⓒ Region

Domain:
- ∃ locatedIn Region H
- locatedIn ∋ ItalianRegion H
- Ⓒ Region
- locatedIn ∋ BourgogneRegi

Inverse Property

□ Symmetric

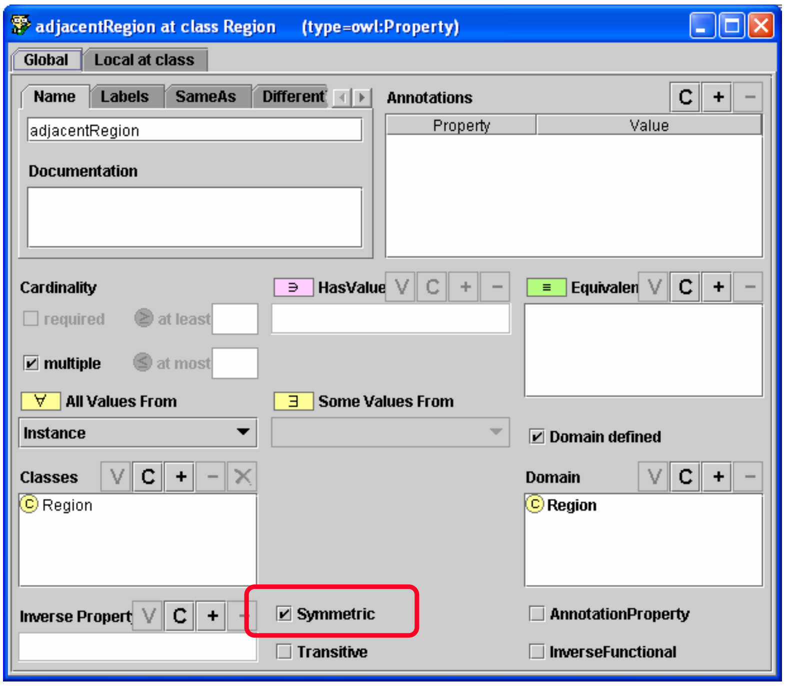
☑ Transitive

□ AnnotationProperty

□ InverseFunctional

Protégé / OWL / Properties / Symmetric

- A symmetric property must always hold in both directions
- MendocinoRegion is adjacent to SonomaRegion and vice-versa.
- Symmetric properties must have equal range and domain



adjacentRegion at class Region (type=owl:Property)

Global | Local at class

Name | Labels | SameAs | Different

adjacentRegion

Documentation

Annotations

Property	Value

Cardinality

□ required ○ at least

☑ multiple ○ at most

∀ All Values From: Instance

∋ HasValue

∃ Some Values From

≡ Equivalent

☑ Domain defined

Classes: Ⓒ Region

Domain: Ⓒ Region

Inverse Property

☑ Symmetric

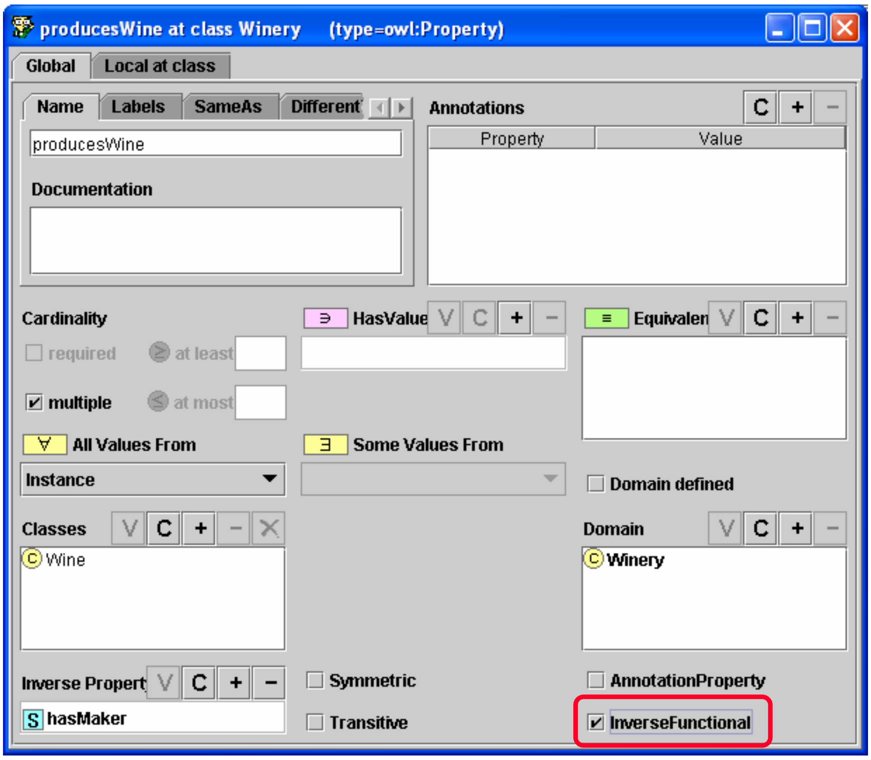
□ Transitive

□ AnnotationProperty

□ InverseFunctional

Protégé / OWL / Properties / InverseFunctional

- For inverse functional properties, if two individuals have the same value then we can infer that the individuals are the same, too
- Example: Each Wine has exactly one Winery as producer



producesWine at class Winery (type=owl:Property)

producesWine

Documentation

Cardinality

□ required at least

☑ multiple at most

Instance

□ Domain defined

Classes: Wine

Domain: Winery

Inverse Propert: hasMaker

□ Symmetric

□ Transitive

□ AnnotationProperty

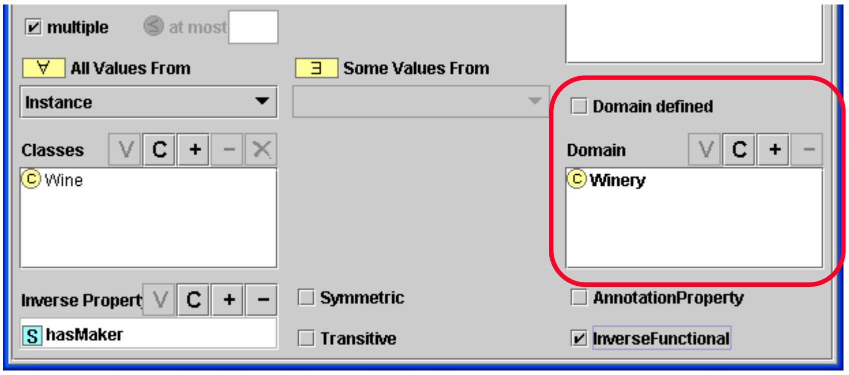
☑ InverseFunctional

Protégé / OWL / Properties / Domain

- The domain of a property is the collection of classes that can have a value for the property
- In Protégé this is traditionally defined from a class point-of-view, i.e. you define a class and then add the properties it can have (the so-called template slots)
- OWL allows to define properties without any restrictions on the domain: This means the property can be used anywhere

→**Difficult to guide form-based knowledge acquisition process**

- Option “Domain defined” specifies whether or not there is a domain restriction defined



☑ multiple at most

Instance

□ Domain defined

Classes: Wine

Domain: Winery

Inverse Propert: hasMaker

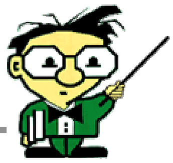
□ Symmetric

□ Transitive

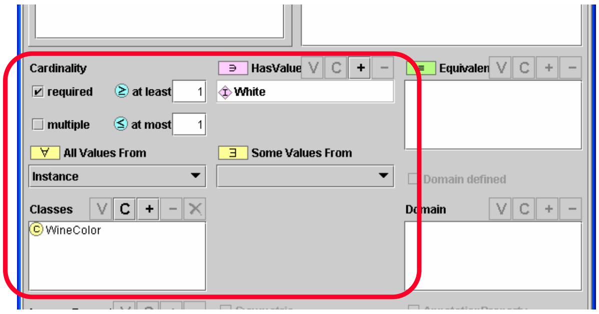
□ AnnotationProperty

☑ InverseFunctional

Protégé / OWL / Properties / Local Characteristics



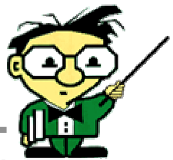
- Many characteristics can be defined for each class individually
- OWL concept: Property Restrictions
- Sometimes called “facet overloading”
- Switch to “Local at class” tab of property



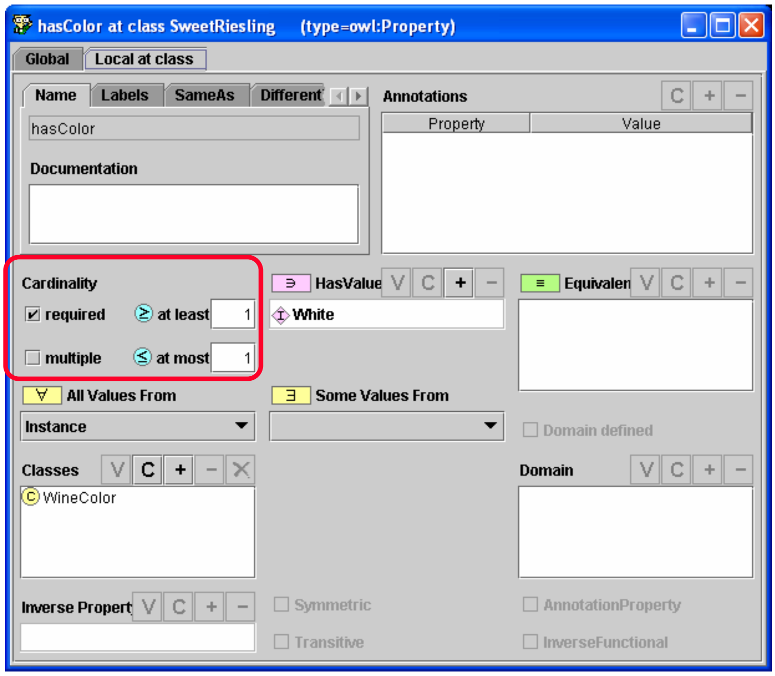
Cardinality
☑ required ≥ at least 1
□ multiple ≤ at most 1
HasValue: White
All Values From: Instance
Some Values From:
Classes: WineColor

	OWL Restriction	Description
Minimum Cardinality	owl:MinimumCardinalityRestriction	The minimum number of values that the property must have at the given class
Maximum Cardinality	owl:MaximumCardinalityRestriction	The maximum number of values that the property can have at the given class
All Values From	owl:AllValuesFrom-Restriction	The type (class) that *all* values of the property must have at the given class
Some Values From	owl:SomeValues-FromRestriction	The type (class) that *at least one* of the values of the property must have at the given class
Has Value	owl:HasValue-Restriction	The value that at least one of the values of the property must be at the given class

Protégé / OWL / Properties / Cardinality



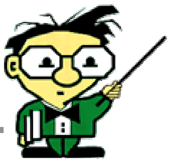
- Cardinality specifies the number of values that the individuals of the class must and can have for the property
  - Minimum cardinality / number of required values
  - Maximum cardinality / number of allowed multiple values



hasColor at class SweetRiesling (type=owl:Property)

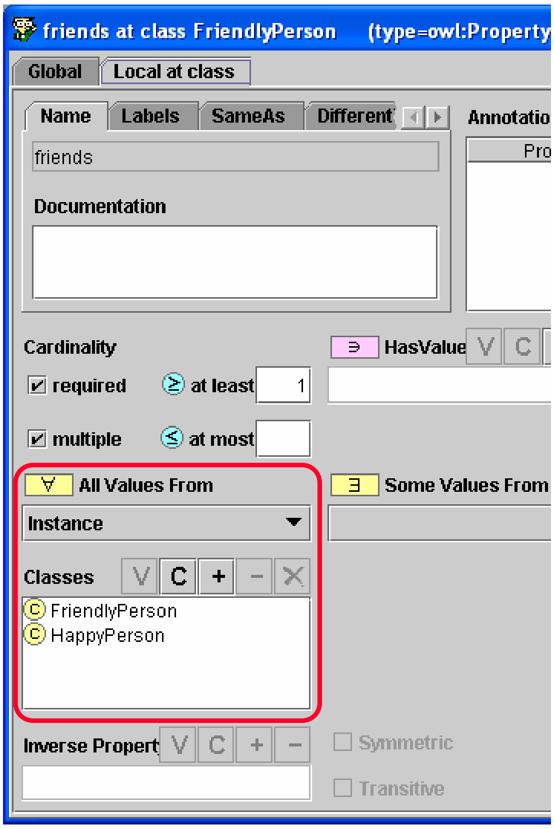
Name: hasColor
Documentation:
Cardinality
☑ required ≥ at least 1
□ multiple ≤ at most 1
HasValue: White
All Values From: Instance
Some Values From:
Classes: WineColor
□ Domain defined
□ Symmetric
□ Transitive
□ AnnotationProperty
□ InverseFunctional

Protégé / OWL / Properties / All Values From



- AllValuesFrom specifies the types that all values of the property must have for instances of this class
- Mostly used for instance properties
- Overloads the global range
- If there is more than one type, then they are interpreted as union, i.e. all values must have one of the types:

```
<owl:allValuesFrom>
  <owl:Class>
    <owl:unionOf rdf:parseType="Collection">
      <owl:Class rdf:about="#FriendlyPerson"/>
      <owl:Class rdf:about="#HappyPerson"/>
    </owl:unionOf>
  </owl:Class>
</owl:allValuesFrom>
```



friends at class FriendlyPerson (type=owl:Property

Name: friends

Documentation

Cardinality

☑ required ≥ at least 1

☑ multiple ≤ at most

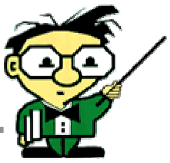
∀ All Values From: Instance

Classes: FriendlyPerson, HappyPerson

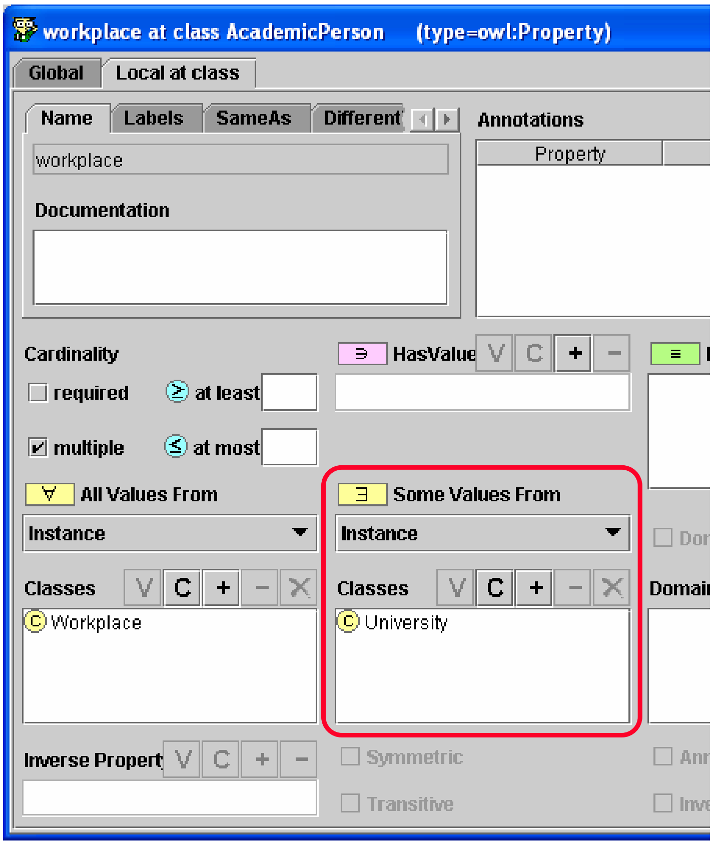
□ Symmetric

□ Transitive

Protégé / OWL / Properties / Some Values From



- Specifies the types that at least one of the values of the property must have for instances of this class
- Mostly used for instance properties
- For example, every AcademicPerson must have at least one workplace at a University



workplace at class AcademicPerson (type=owl:Property)

Name: workplace

Documentation

Cardinality

□ required ≥ at least

☑ multiple ≤ at most

∀ All Values From: Instance

Classes: Workplace

∃ Some Values From: Instance

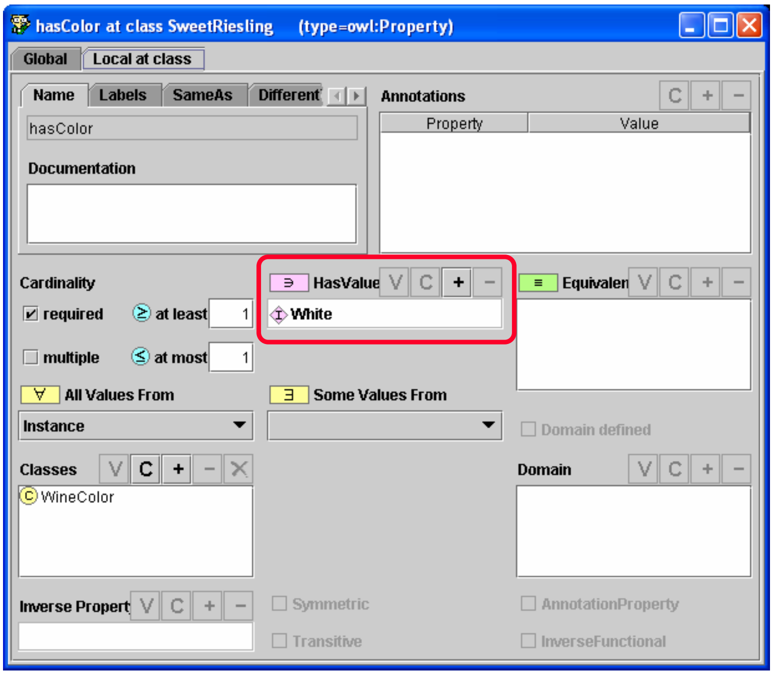
Classes: University

□ Symmetric

□ Transitive

Protégé / OWL / Properties / Has Value

- Specifies one value that the individuals of the class must have for the property
- For example, all Sweet Rieslings have white color



hasColor at class SweetRiesling (type=owl:Property)

Name: hasColor

Cardinality: ☑ required, at least 1; □ multiple, at most 1

HasValue: White

All Values From: Instance

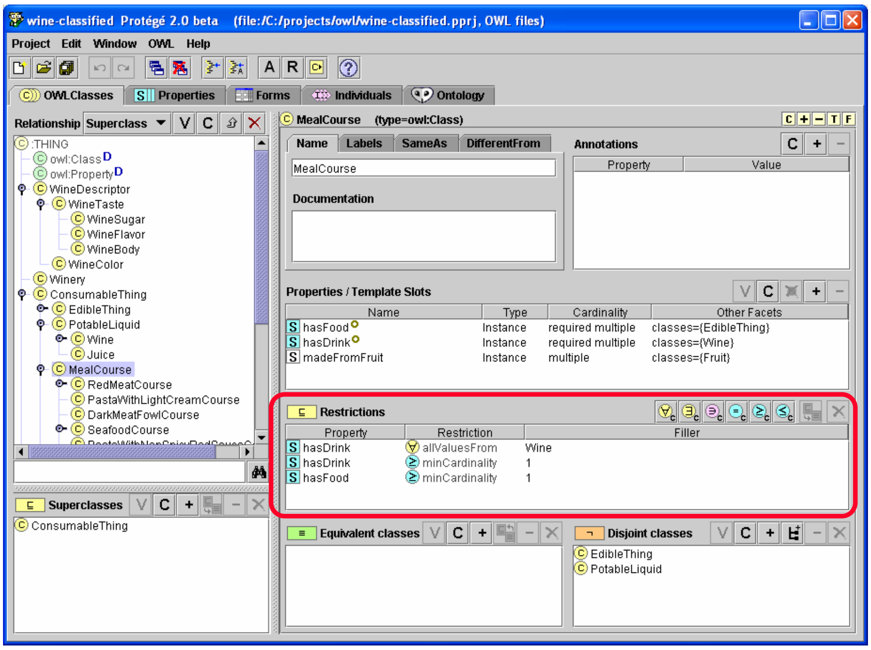
Classes: WineColor

□ Domain defined

□ Symmetric □ Transitive □ AnnotationProperty □ InverseFunctional

Protégé / OWL / Restrictions

- Restrictions are a special type of anonymous class definitions
- A restriction defines an anonymous class
- Members of this class are all individuals that fulfill the restriction
- Restrictions are often used to define necessary conditions for a class



wine-classified Protégé 2.0 beta (file:/C:/projects/owl/wine-classified.pprj, OWL files)

```
:THING
  owl:Class D
  owl:Property D
  WineDescriptor
    WineTaste
      WineSugar
      WineFlavor
      WineBody
    WineColor
  Winery
  ConsumableThing
    EdibleThing
    PotableLiquid
      Wine
      Juice
    MealCourse
      RedMeatCourse
      PastaWithLightCreamCourse
      DarkMeatFowlCourse
      SeafoodCourse
```

MealCourse (type=owl:Class)

Name: MealCourse

Properties / Template Slots

Name	Type	Cardinality	Other Facets
hasFood	Instance	required multiple	classes={EdibleThing}
hasDrink	Instance	required multiple	classes={Wine}
madeFromFruit	Instance	multiple	classes={Fruit}

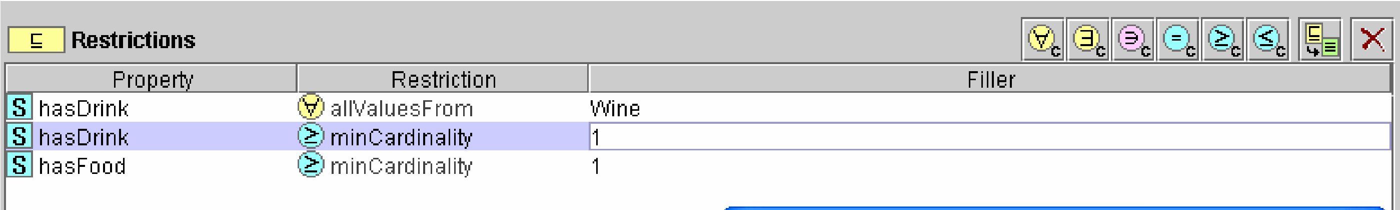
Restrictions

Property	Restriction	Filler
hasDrink	allValuesFrom	Wine
hasDrink	minCardinality	1
hasFood	minCardinality	1

Superclasses: ConsumableThing

Disjoint classes: EdibleThing, PotableLiquid

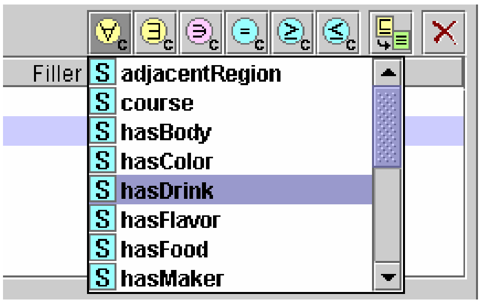
Protégé / OWL / Restrictions / Editing



Restrictions

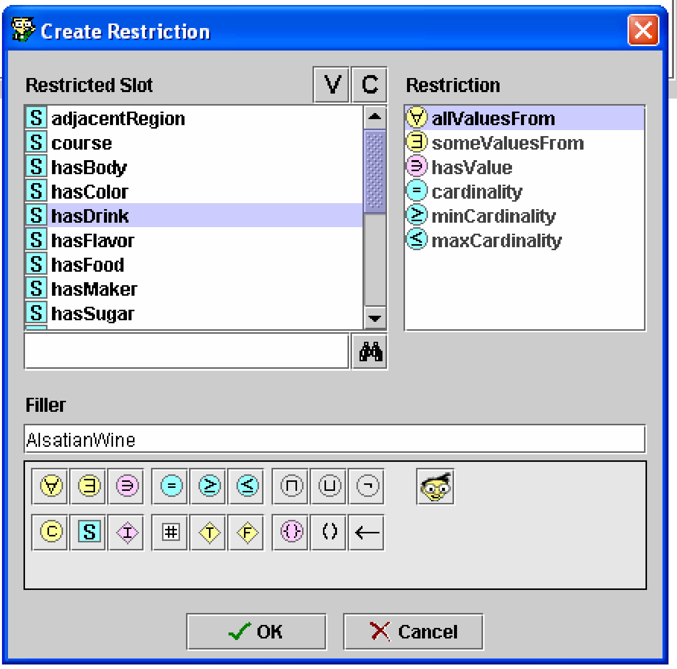
Property	Restriction	Filler
S hasDrink	allValuesFrom	Wine
S hasDrink	minCardinality	1
S hasFood	minCardinality	1

- To add a restriction either
  - Left-click one of the c buttons; this will open the dialog on the right
  - Right-click one of the c buttons, select the property to restrict and then edit the filler in the table



Filler

S adjacentRegion
S course
S hasBody
S hasColor
S hasDrink
S hasFlavor
S hasFood
S hasMaker



Create Restriction

Restricted Slot

S adjacentRegion
S course
S hasBody
S hasColor
S hasDrink
S hasFlavor
S hasFood
S hasMaker
S hasSugar

Restriction

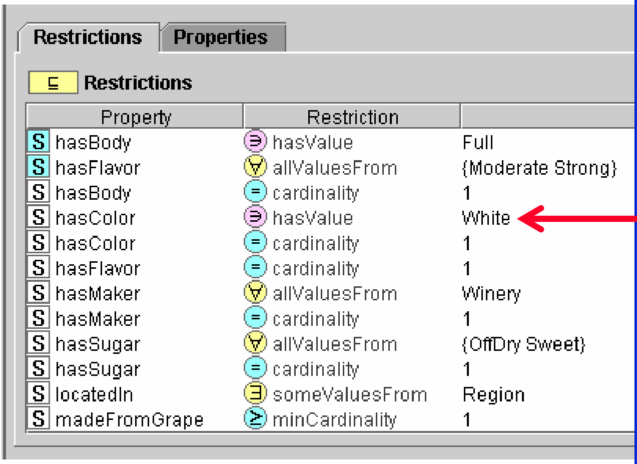
allValuesFrom
someValuesFrom
hasValue
cardinality
minCardinality
maxCardinality

Filler

AlsatianWine

Protégé / OWL / Restrictions / Synchronization (1)

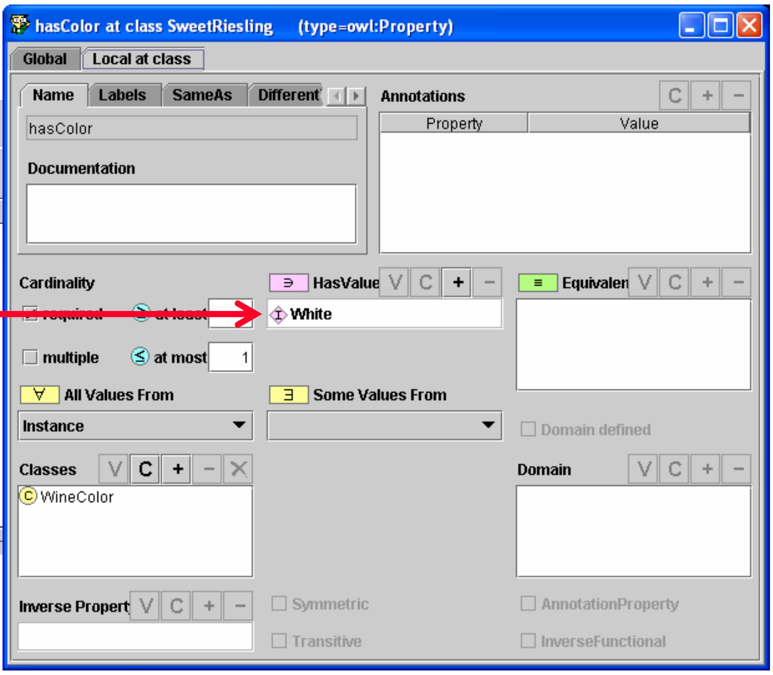
- Local characteristics and restrictions are synchronized
- For example, if you state that SweetRiesling has white color, then the system automatically adds a HasValue restriction on the property for the class



Restrictions | Properties

Restrictions

Property	Restriction	
S hasBody	hasValue	Full
S hasFlavor	allValuesFrom	{Moderate Strong}
S hasBody	cardinality	1
S hasColor	hasValue	White
S hasColor	cardinality	1
S hasFlavor	cardinality	1
S hasMaker	allValuesFrom	Winery
S hasMaker	cardinality	1
S hasSugar	allValuesFrom	{OffDry Sweet}
S hasSugar	cardinality	1
S locatedIn	someValuesFrom	Region
S madeFromGrape	minCardinality	1



hasColor at class SweetRiesling (type=owl:Property)

Global | Local at class

Name | Labels | SameAs | Different

hasColor

Documentation

Annotations: Property | Value

Cardinality

☑ required at least

□ multiple at most 1

HasValue: White

Equivalent

All Values From: Instance

Some Values From

□ Domain defined

Classes: WineColor

Domain

Inverse Property

□ Symmetric

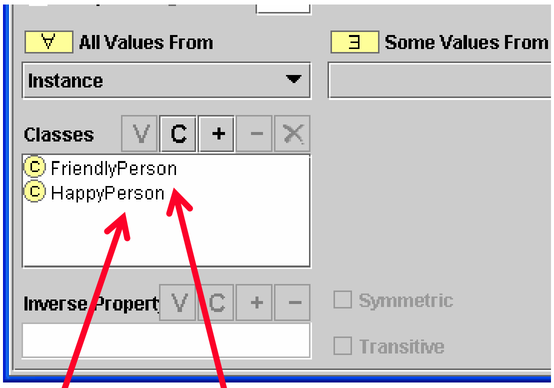
□ Transitive

□ AnnotationProperty

□ InverseFunctional

Protégé / OWL / Restrictions / Synchronization (2)

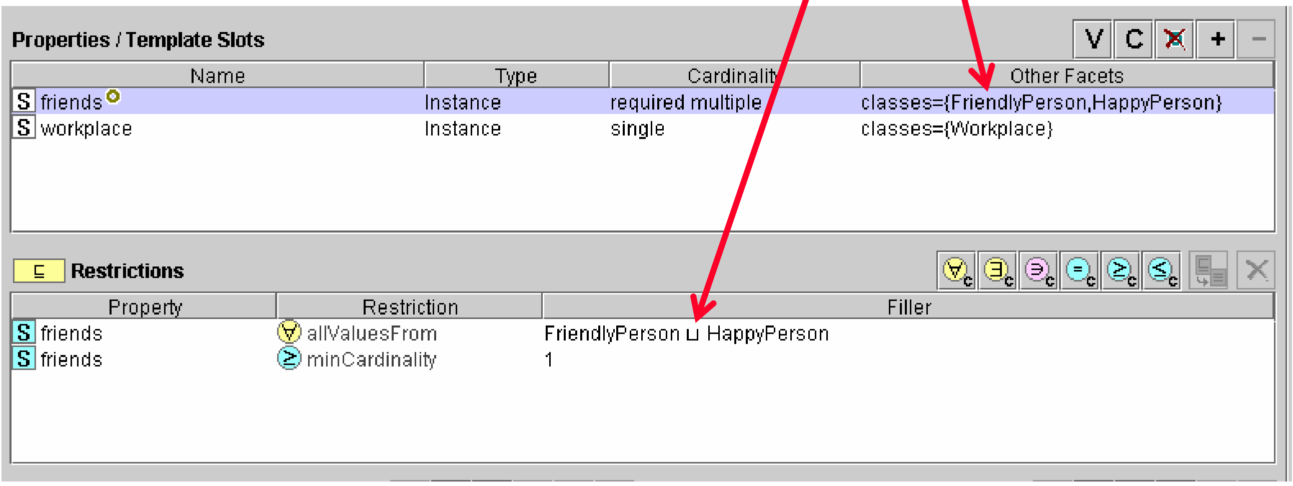
- In this example, multiple allowed classes for the property friends are converted into an allValuesFrom restriction at the class with a union of the two allowed classes



FriendlyPerson
HappyPerson

□ Symmetric

□ Transitive



Name	Type	Cardinality	Other Facets
S friends	Instance	required multiple	classes={FriendlyPerson,HappyPerson}
S workplace	Instance	single	classes={Workplace}

Property	Restriction	Filler
S friends	∀ allValuesFrom	FriendlyPerson ⊔ HappyPerson
S friends	≥ minCardinality	1

Protégé / OWL / Restrictions / Synchronization (3)

- Two mechanisms to edit restrictions:

- Define local characteristics on the property form

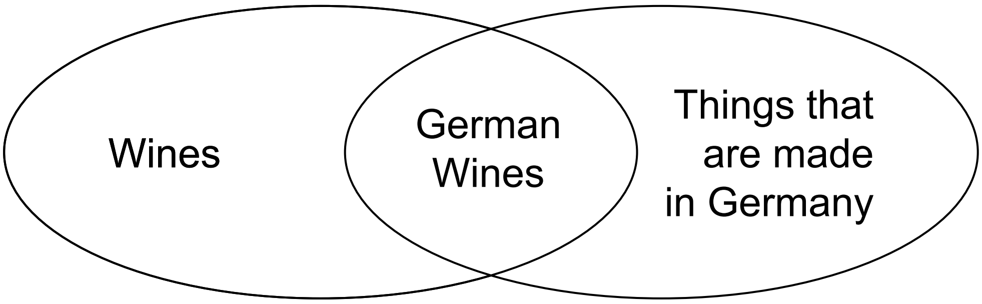
  → **Bundles all characteristics from a property’s point of view**

- Define restrictions on the class form

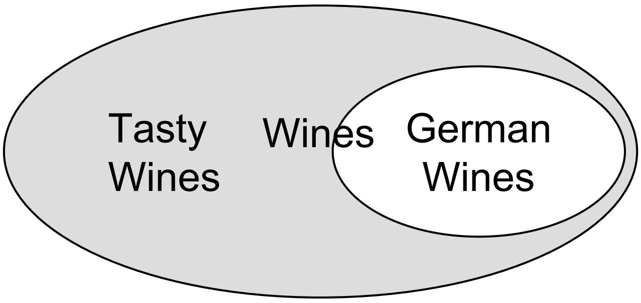
  → **Displays the class definition**

Protégé / OWL / Expressions

- OWL supports class definitions made out of logical combinations of other classes
  - Example: “A GermanWine is a Wine that is made in Germany”



  - Example: “A tasty Wine is a Wine that is not a German Wine”



Protégé / OWL / Expression Syntax

- RDF is the official OWL syntax:

```
<owl:Class rdf:ID="GermanWine">
    <owl:equivalentClass>
      <owl:Class>
        <owl:intersectionOf rdf:parseType="Collection">
          <owl:Class rdf:about="#Wine"/>
          <owl:Restriction>
            <owl:onProperty>
              <owl:ObjectProperty rdf:about="#locatedIn"/>
            </owl:onProperty>
            <owl:hasValue rdf:resource="#GermanyRegion" rdf:type="#Region"/>
          </owl:Restriction>
        </owl:intersectionOf>
      </owl:Class>
    </owl:equivalentClass>
    <rdfs:subClassOf>
      <owl:Class rdf:about="#Wine"/>
    </rdfs:subClassOf>
  </owl:Class>
```

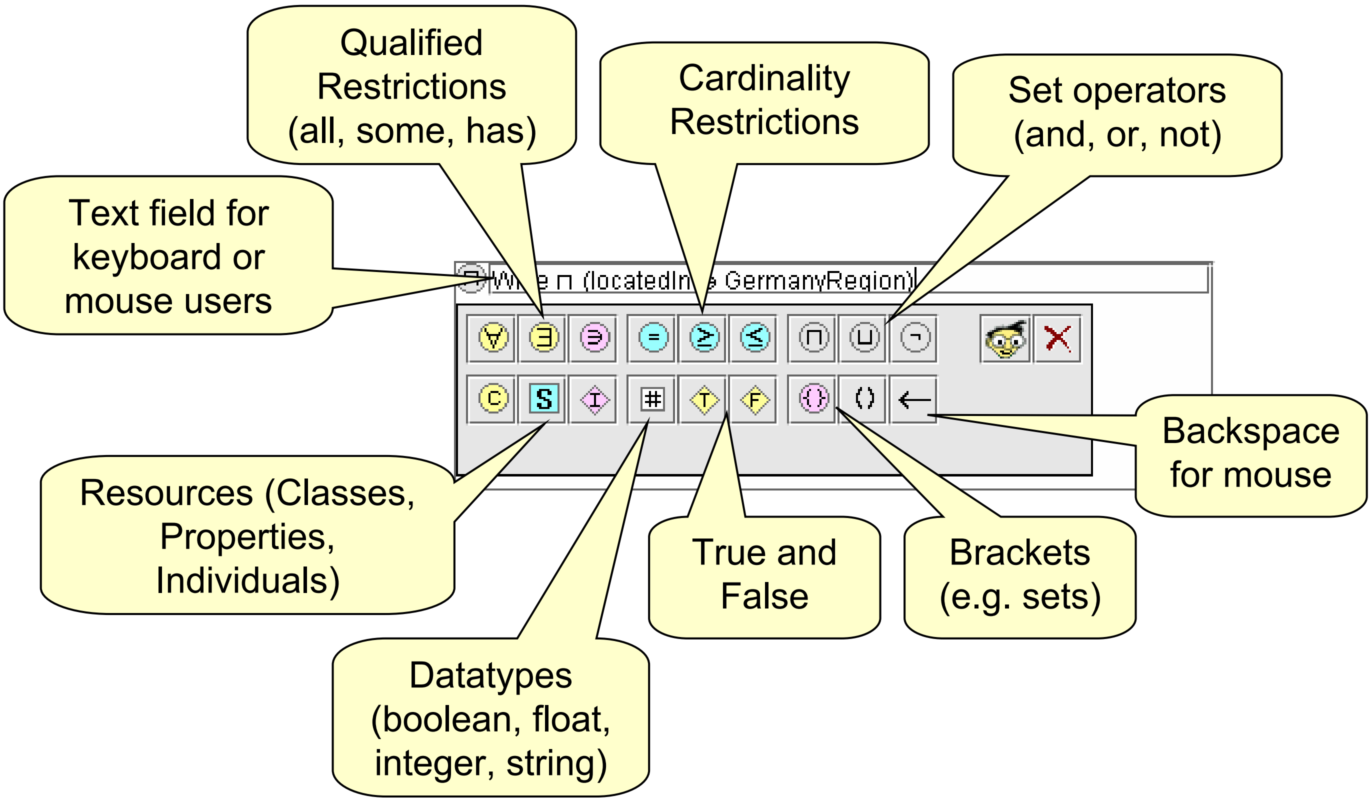
- Intended for machines
- Needed: A user friendly syntax

$\text{Wine} \sqcap (\text{locatedIn} \ni \text{GermanyRegion})$

Protégé / OWL / Expression Syntax in Protégé

OWL Element	Symbol	Key	Example
allValuesFrom	∀	*	∀ children Male
someValuesFrom	∃	?	∃ children Lawyer
hasValue	∋	$	rich ∋ true
cardinality	=	=	children = 3
minCardinality	≥	>	children ≥ 3
maxCardinality	≤	<	children ≤ 3
complementOf	¬	!	¬ Parent
intersectionOf	⊓	&	Human ⊓ Male
unionOf	⊔	|	Doctor ⊔ Lawyer
enumeration	{...}	{ }	{male female}

Protégé / OWL / Expression Editor



Qualified Restrictions (all, some, has)

Cardinality Restrictions

Set operators (and, or, not)

Text field for keyboard or mouse users

Backspace for mouse

Resources (Classes, Properties, Individuals)

True and False

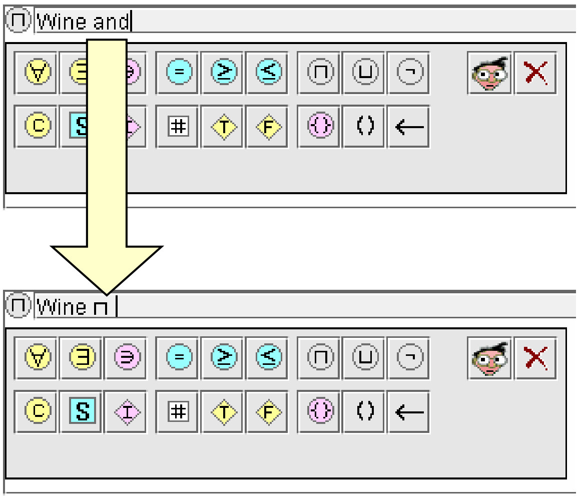
Brackets (e.g. sets)

Datatypes (boolean, float, integer, string)

Protégé / OWL / Expression Editor / Synonyms

- Some symbols are not on the keyboard
- There are special keys for them, but also synonyms (after SPACE)

Symbol	Key	Synonyms
∀	*	all allValuesFrom forall only
∃	?	some someValuesFrom exists
∋	$	has hasValue value
=	=	
≥	>	
≤	<	
¬	!	not
⊓	&	and
⊔	|	or
{...}	{ }	



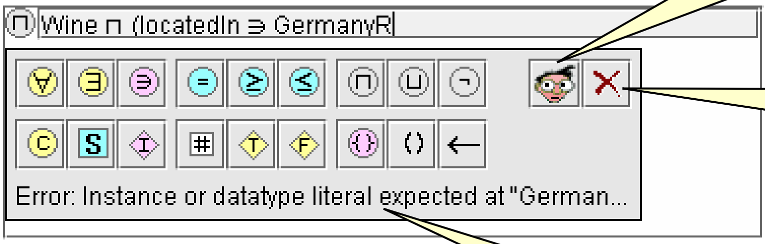
Wine and

Wine ⊓

Protégé / OWL / Expression Editor / Syntax Checking

- You can only assign valid OWL expressions
- The Editor checks the syntax while you type
- The “nerd” icon will get a red face if something is wrong
- Pressing ENTER then displays an error message

Error indicator and OK button



Wine ⊓ (locatedIn ∋ GermanyR

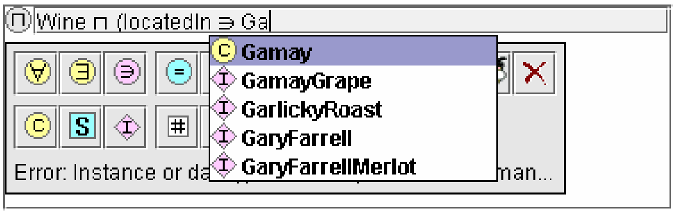
Error: Instance or datatype literal expected at "German...

Cancel changes

Error display

Protégé / OWL / Expression Editor / Auto-Completion

- Keyboard users only need to type the first few letters of a class/property/individual name
- Pressing TAB or CTRL+SPACE opens Auto-Completion box
- If there is only one choice, the name is inserted directly
- Select an existing name or continue typing to narrow/widen the choice
- UP/DOWN to select, ENTER to insert, ESCAPE to cancel



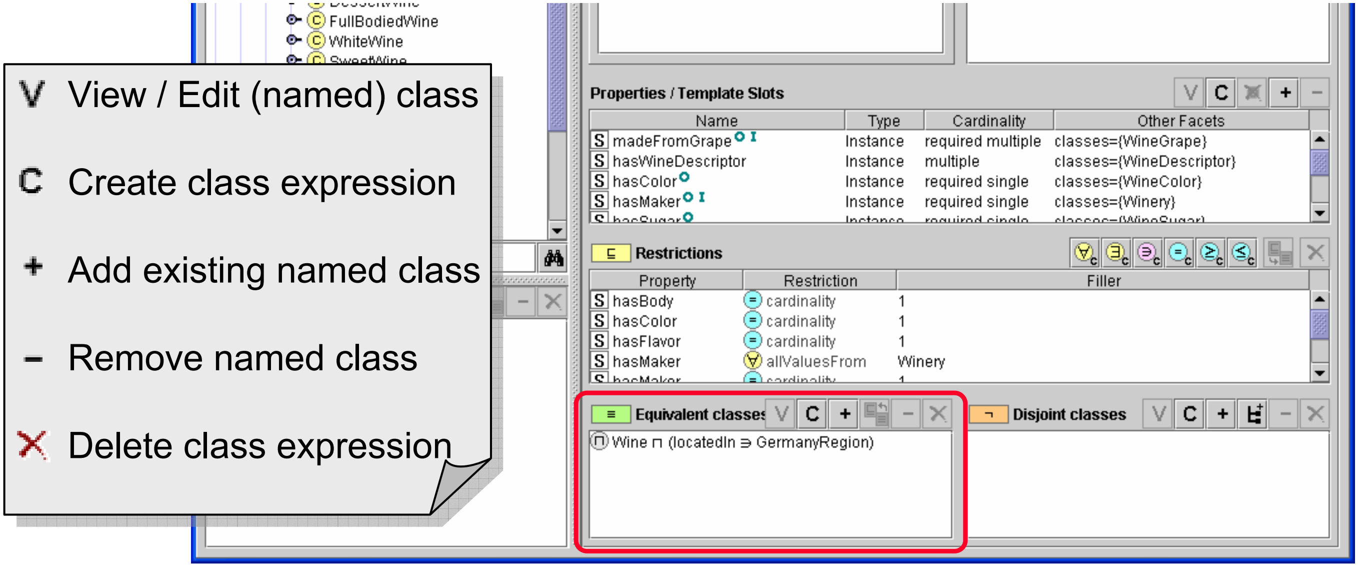
Wine ⊓ (locatedIn ∋ Ga

- Gamay
- GamayGrape
- GarlickyRoast
- GaryFarrell
- GaryFarrellMerlot

Error: Instance or da

Protégé / OWL / Classes / Equivalent Classes

- List of classes that are known to have exactly the same instances
- Equivalent classes are super/sub classes of each other
- Can contain named classes or class expressions
- Defines necessary and required conditions for class membership
- Can be used as logical definition of the class



V View / Edit (named) class

C Create class expression

+ Add existing named class

– Remove named class

✕ Delete class expression

Properties / Template Slots

Name	Type	Cardinality	Other Facets
S madeFromGrape	Instance	required multiple	classes={WineGrape}
S hasWineDescriptor	Instance	multiple	classes={WineDescriptor}
S hasColor	Instance	required single	classes={WineColor}
S hasMaker	Instance	required single	classes={Winery}

Restrictions

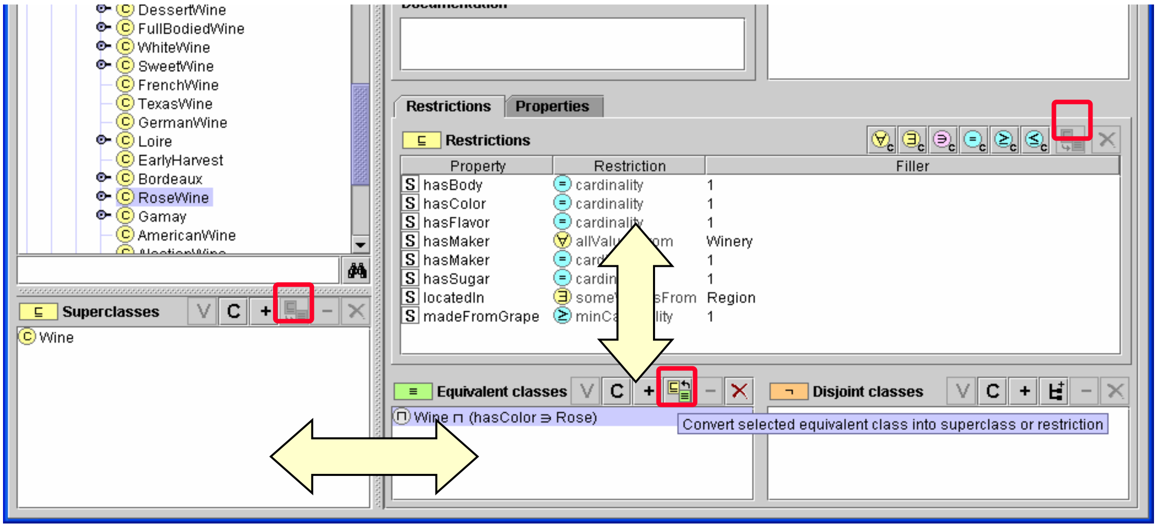
Property	Restriction	Filler
S hasBody	cardinality	1
S hasColor	cardinality	1
S hasFlavor	cardinality	1
S hasMaker	allValuesFrom	Winery

Equivalent classes

Wine ⊓ (locatedIn ∋ GermanyRegion)

Protégé / OWL / Classes / Moving Classes

- Superclasses / Restrictions define necessary conditions
- Equivalent classes define necessary and sufficient conditions
- Sometimes during ontology evolution, users may want to switch
- Equivalent classes are moved into
  - Restrictions: if they are restrictions
  - "Plain" superclasses: otherwise



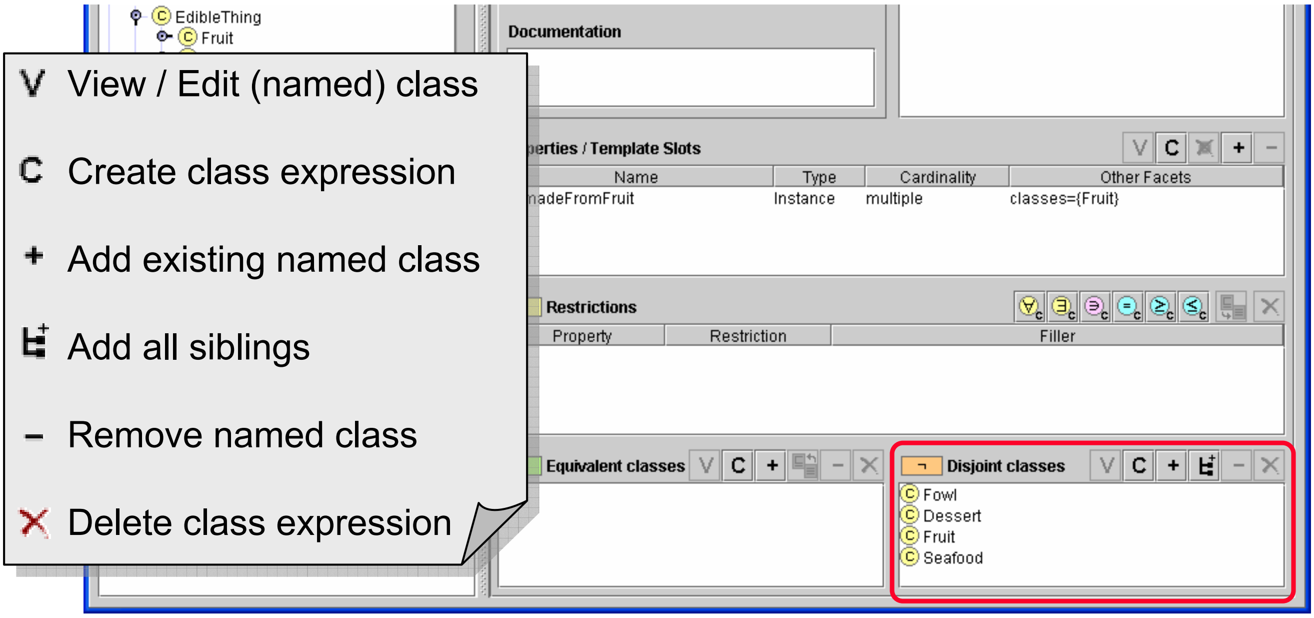
Property	Restriction	Filler
hasBody	cardinality	1
hasColor	cardinality	1
hasFlavor	cardinality	1
hasMaker	allValu om	Winery
hasMaker	card	1
hasSugar	cardin	1
locatedIn	some sFrom	Region
madeFromGrape	minCa lity	1

Wine ⊓ (hasColor ∋ Rose)

Convert selected equivalent class into superclass or restriction

Protégé / OWL / Classes / Disjoint Classes

- In OWL, classes may have shared instances
- If two classes are defined as disjoint then there is no individual that is member of both at the same time
- Protégé usually adds both directions of this symmetric relationship



V View / Edit (named) class

C Create class expression

+ Add existing named class

Add all siblings

– Remove named class

✕ Delete class expression

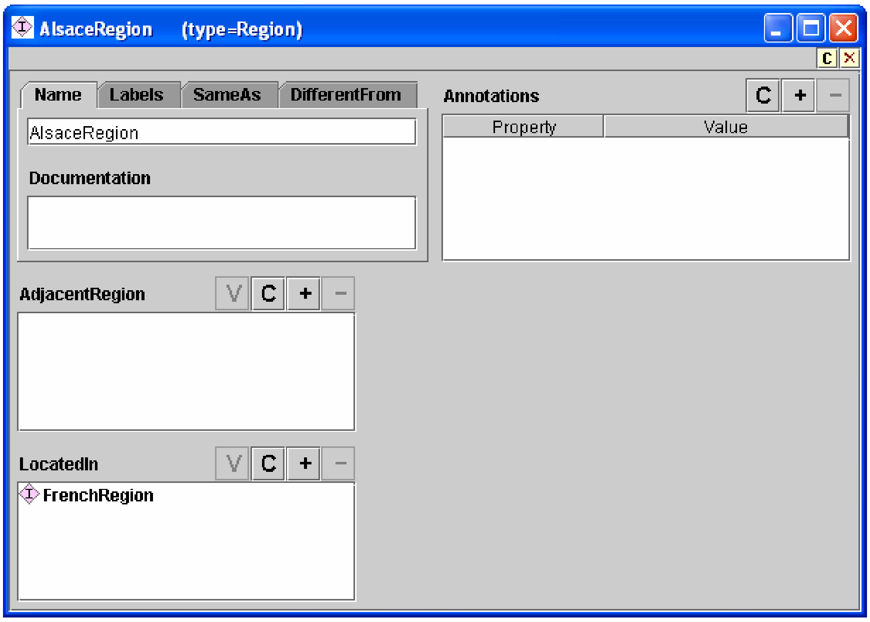
perties / Template Slots

Name	Type	Cardinality	Other Facets
nadeFromFruit	Instance	multiple	classes={Fruit}

Property	Restriction	Filler

Protégé / OWL / Individuals

- Individuals are specific instances of the classes from the ontology
- Each individual can have multiple classes as “types”
- Protégé automatically creates forms for the acquisition of individuals



AlsaceRegion (type=Region)

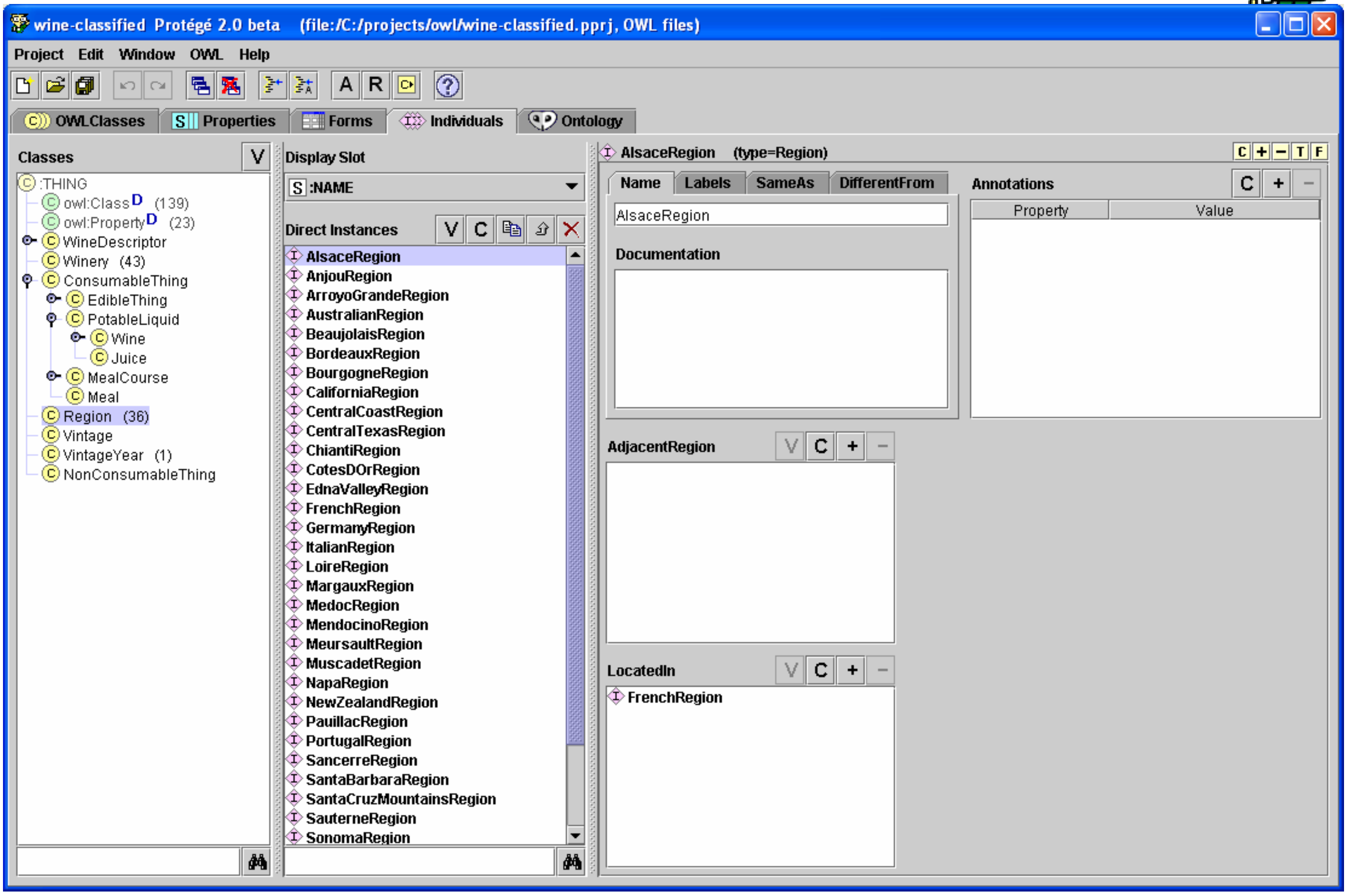
Name: AlsaceRegion

Documentation:

AdjacentRegion:

LocatedIn: FrenchRegion

Protégé / OWL / Individuals / Tab



wine-classified Protégé 2.0 beta (file:/C:/projects/owl/wine-classified.pprj, OWL files)

Classes:
- :THING
  - owl:Class (139)
  - owl:Property (23)
  - WineDescriptor
  - Winery (43)
  - ConsumableThing
    - EdibleThing
    - PotableLiquid
      - Wine
      - Juice
    - MealCourse
    - Meal
  - Region (36)
  - Vintage
  - VintageYear (1)
  - NonConsumableThing

Display Slot: :NAME

Direct Instances:
- AlsaceRegion
- AnjouRegion
- ArroyoGrandeRegion
- AustralianRegion
- BeaujolaisRegion
- BordeauxRegion
- BourgogneRegion
- CaliforniaRegion
- CentralCoastRegion
- CentralTexasRegion
- ChiantiRegion
- CotesDOrRegion
- EdnaValleyRegion
- FrenchRegion
- GermanyRegion
- ItalianRegion
- LoireRegion
- MargauxRegion
- MedocRegion
- MendocinoRegion
- MeursaultRegion
- MuscadetRegion
- NapaRegion
- NewZealandRegion
- PauillacRegion
- PortugalRegion
- SancerreRegion
- SantaBarbaraRegion
- SantaCruzMountainsRegion
- SauterneRegion
- SonomaRegion

AlsaceRegion (type=Region)

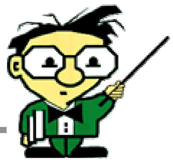
Name: AlsaceRegion

Documentation:

AdjacentRegion:

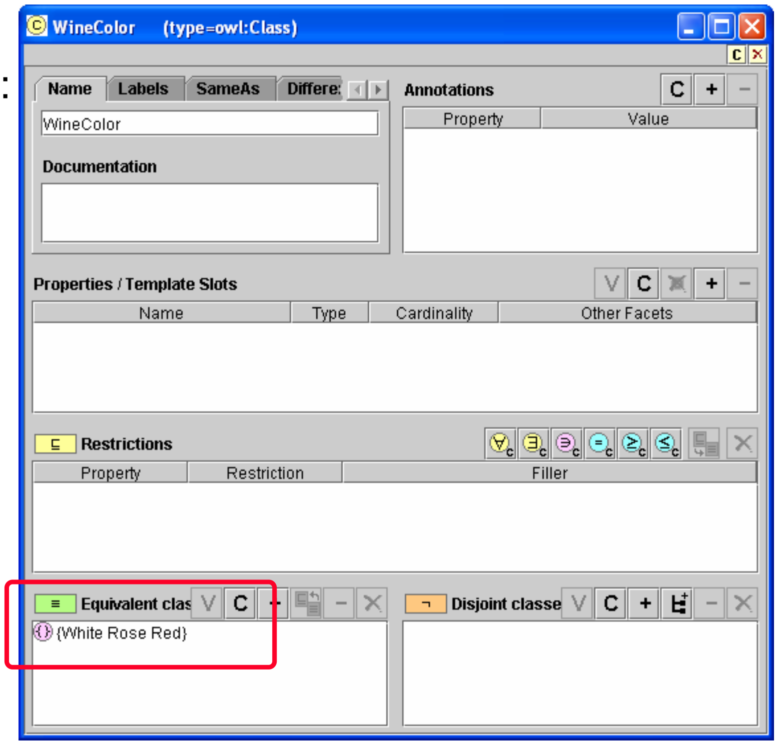
LocatedIn: FrenchRegion

Protégé / OWL / Individuals / Enumerated classes

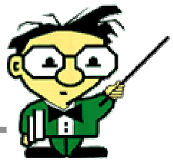


- Enumerated classes are defined by listing all its instances exhaustively
- Often used in an equivalent class statement

- To define an enumerated class:
  1. Define the class
  2. Create the individuals (e.g. White) using the Individuals Tab
  3. Create an equivalent enumeration using { ... }

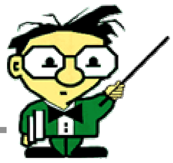


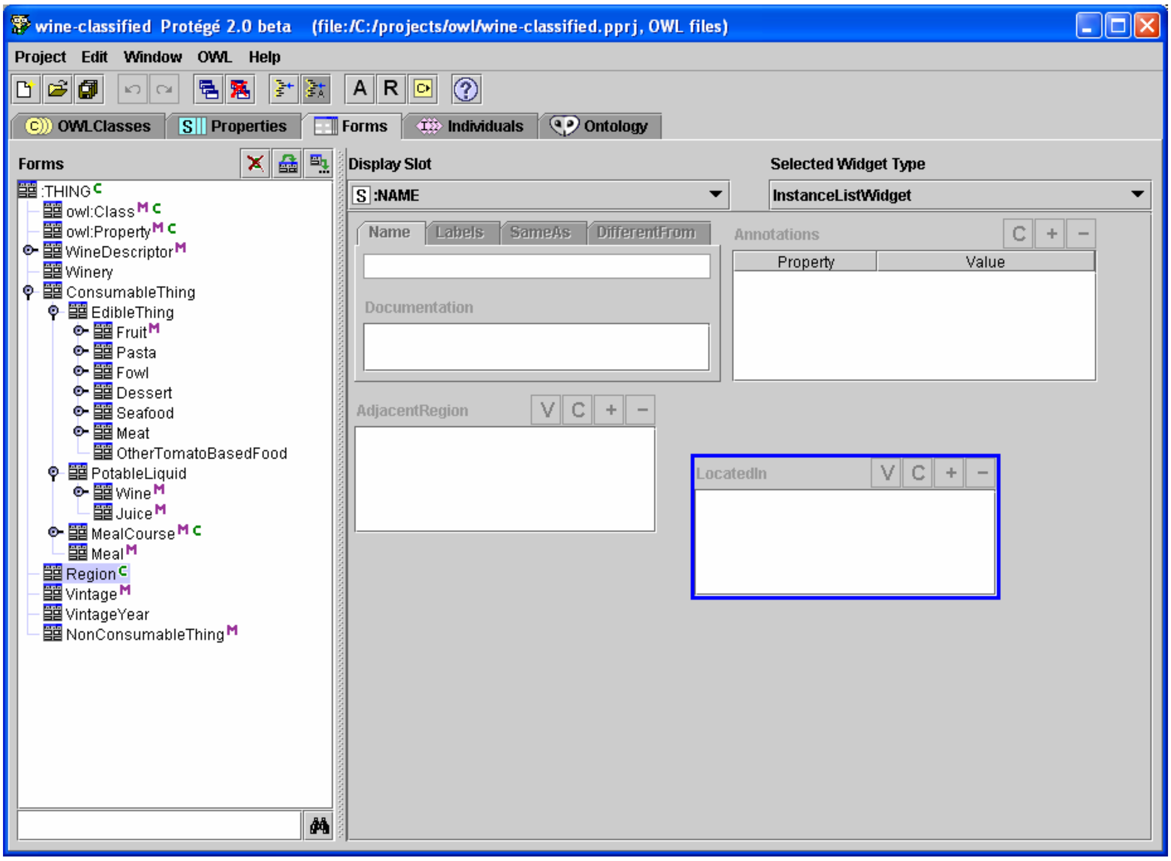
Protégé / OWL / Forms Tab



- Protégé automatically generates forms for all classes
- Each property is represented by one widget
- Default widgets are selected for the value types
- Default layout heuristics are applied for user-defined classes
- The default layout may be sufficient in most cases but often applications want to make custom-tailored adaptations
- The Forms Tab can be used
  - to modify the layout of the forms
  - to select alternative widgets for selected properties

Protégé / OWL / Forms / Tab



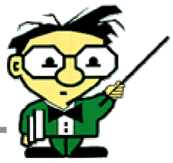


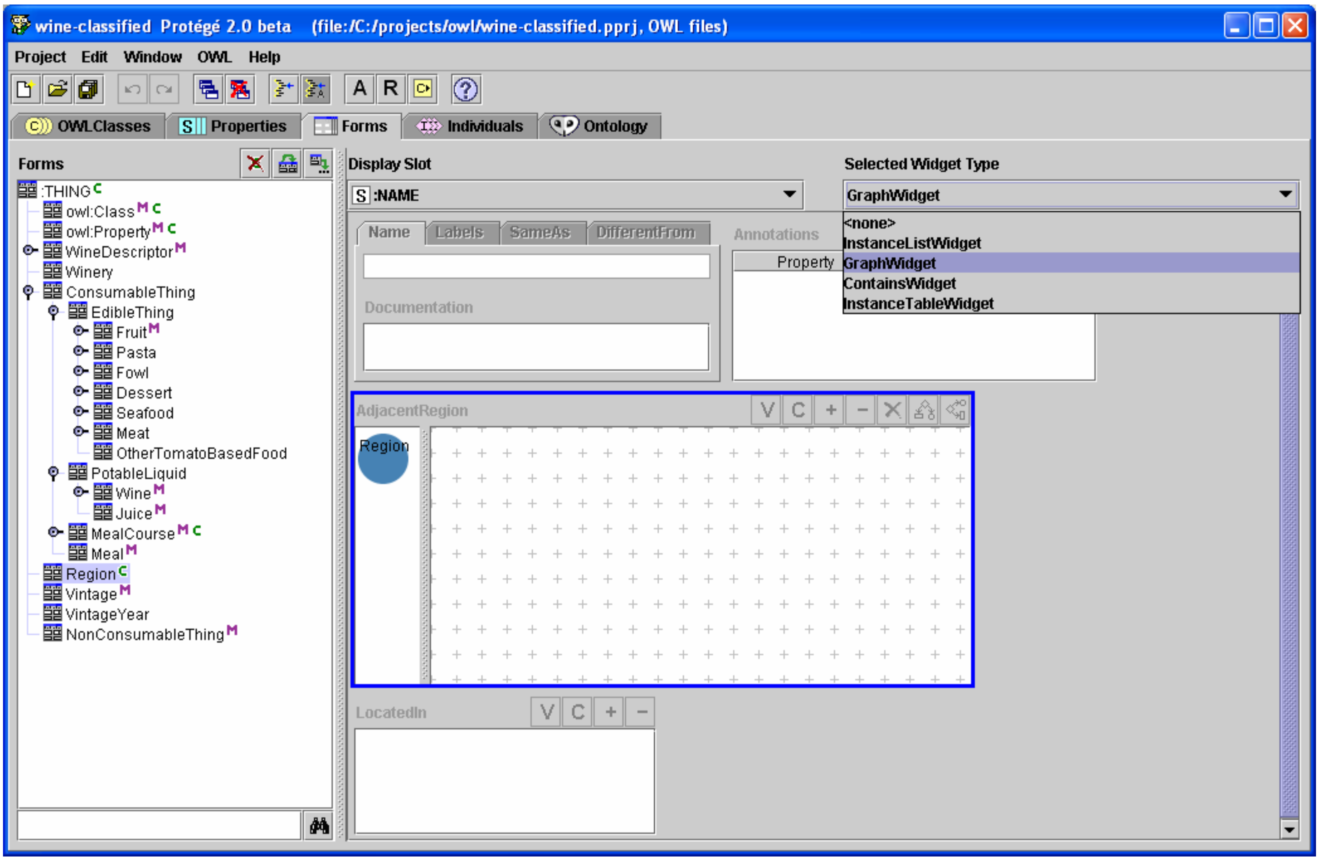
- :THING C
  - owl:Class M C
  - owl:Property M C
  - WineDescriptor M
  - Winery
  - ConsumableThing
    - EdibleThing
      - Fruit M
      - Pasta
      - Fowl
      - Dessert
      - Seafood
      - Meat
      - OtherTomatoBasedFood
    - PotableLiquid
      - Wine M
      - Juice M
    - MealCourse M C
    - Meal M
  - Region C
  - Vintage M
  - VintageYear
  - NonConsumableThing M

Display Slot: :NAME

Selected Widget Type: InstanceListWidget

Protégé / OWL / Forms / Selecting Widgets





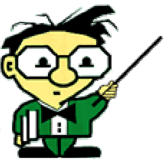
- :THING C
  - owl:Class M C
  - owl:Property M C
  - WineDescriptor M
  - Winery
  - ConsumableThing
    - EdibleThing
      - Fruit M
      - Pasta
      - Fowl
      - Dessert
      - Seafood
      - Meat
      - OtherTomatoBasedFood
    - PotableLiquid
      - Wine M
      - Juice M
    - MealCourse M C
    - Meal M
  - Region C
  - Vintage M
  - VintageYear
  - NonConsumableThing M

Display Slot: :NAME

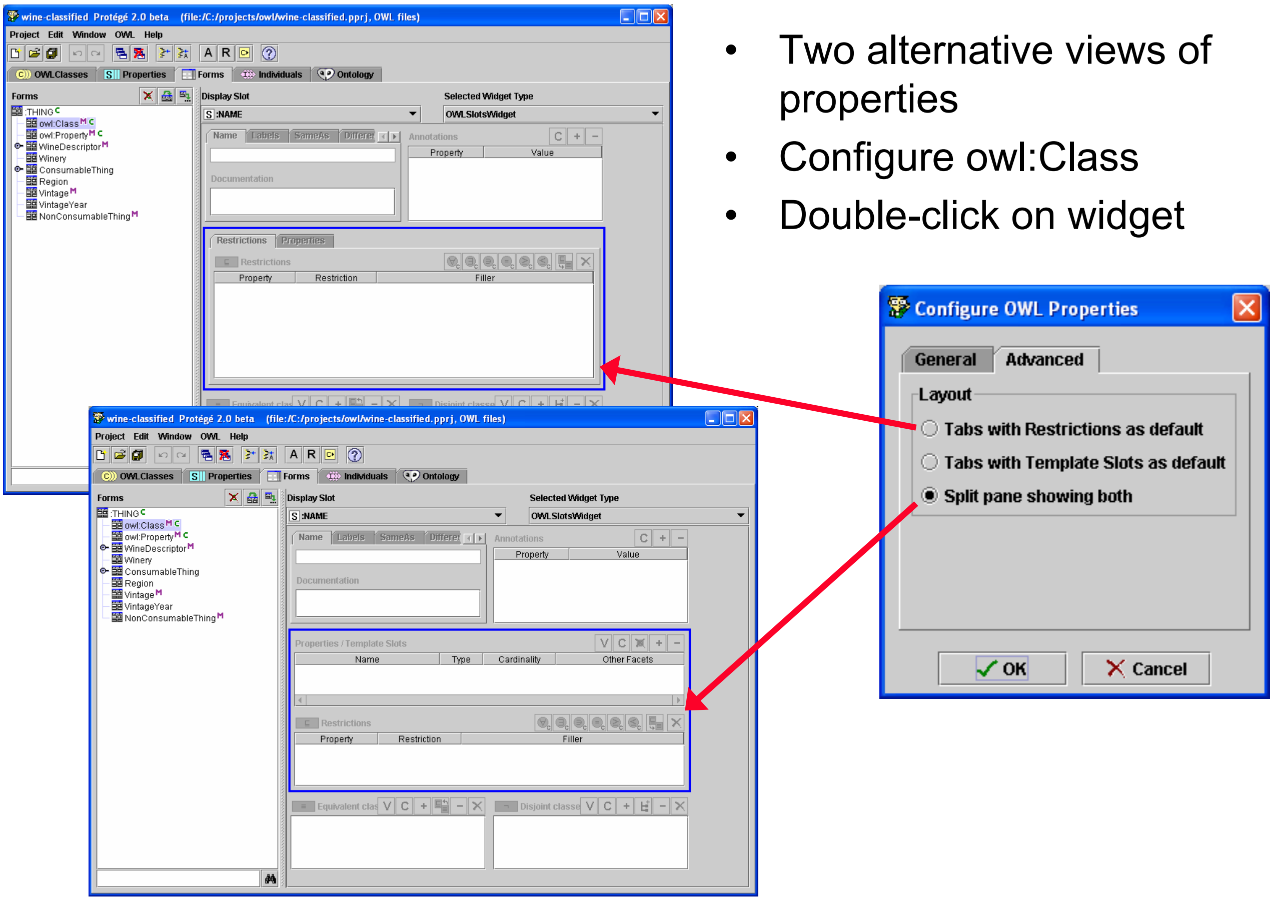
Selected Widget Type: GraphWidget

- <none>
- InstanceListWidget
- GraphWidget
- ContainsWidget
- InstanceTableWidget

Protégé / OWL / Forms / Editing the class form



- Two alternative views of properties
- Configure owl:Class
- Double-click on widget



Configure OWL Properties

General | Advanced

Layout

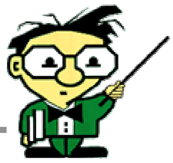
○ Tabs with Restrictions as default

○ Tabs with Template Slots as default

● Split pane showing both

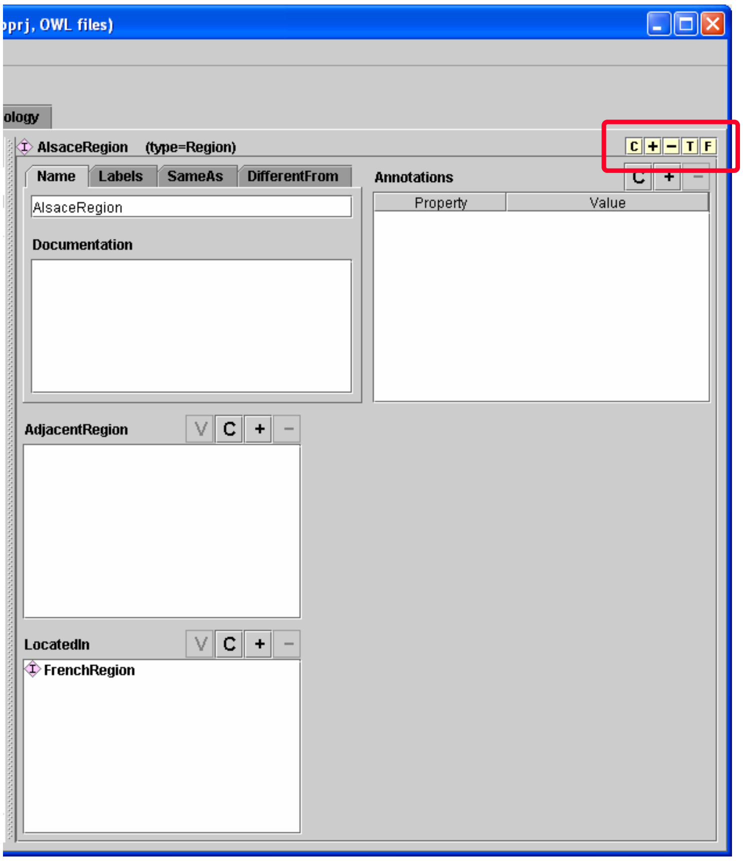
✓ OK | ✗ Cancel

Protégé / OWL / Forms / Adding properties quickly



- The small buttons allow users to change the type of a resource, and its form, rapidly, without having to switch tabs
- Allow simple building of "anonymous types" to ask queries

C	Create new property for type
+	Add existing property to type
–	Remove property from type
T	Edit type (opens class form)
F	Edit form in Forms Tab



AlsaceRegion (type=Region)

Name: AlsaceRegion

Documentation

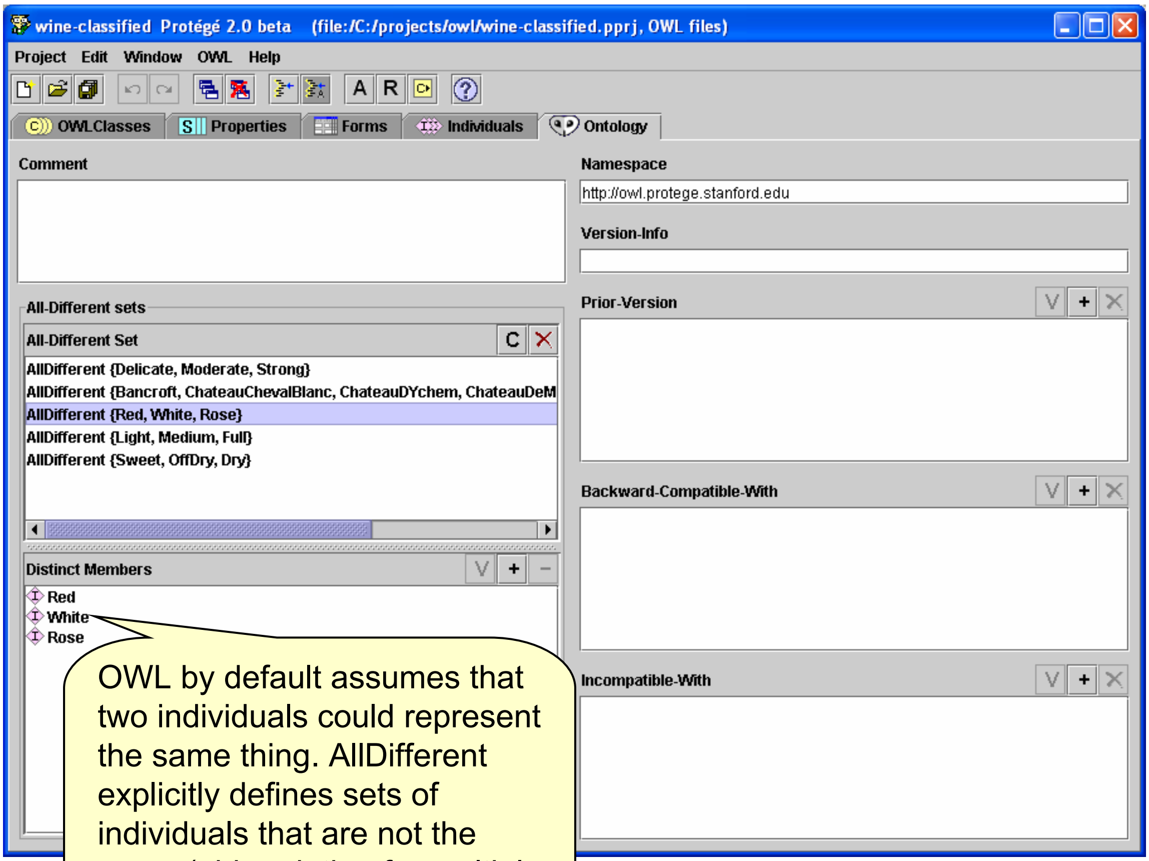
Annotations

Property	Value

AdjacentRegion

LocatedIn: FrenchRegion

Protégé / OWL / Ontology Metadata



wine-classified Protégé 2.0 beta (file:/C:/projects/owl/wine-classified.pprj, OWL files)

Comment:

Namespace: http://owl.protege.stanford.edu

Version-Info:

All-Different sets

All-Different Set

AllDifferent {Delicate, Moderate, Strong}
AllDifferent {Bancroft, ChateauChevalBlanc, ChateauDYchem, ChateauDeM
AllDifferent {Red, White, Rose}
AllDifferent {Light, Medium, Full}
AllDifferent {Sweet, OffDry, Dry}

Distinct Members

Red
White
Rose

Prior-Version:

Backward-Compatible-With:

Incompatible-With:

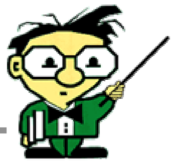
OWL by default assumes that two individuals could represent the same thing. AllDifferent explicitly defines sets of individuals that are not the same (abbreviation for multiple owl:differentFrom statements).

Protégé / OWL / Species Validator

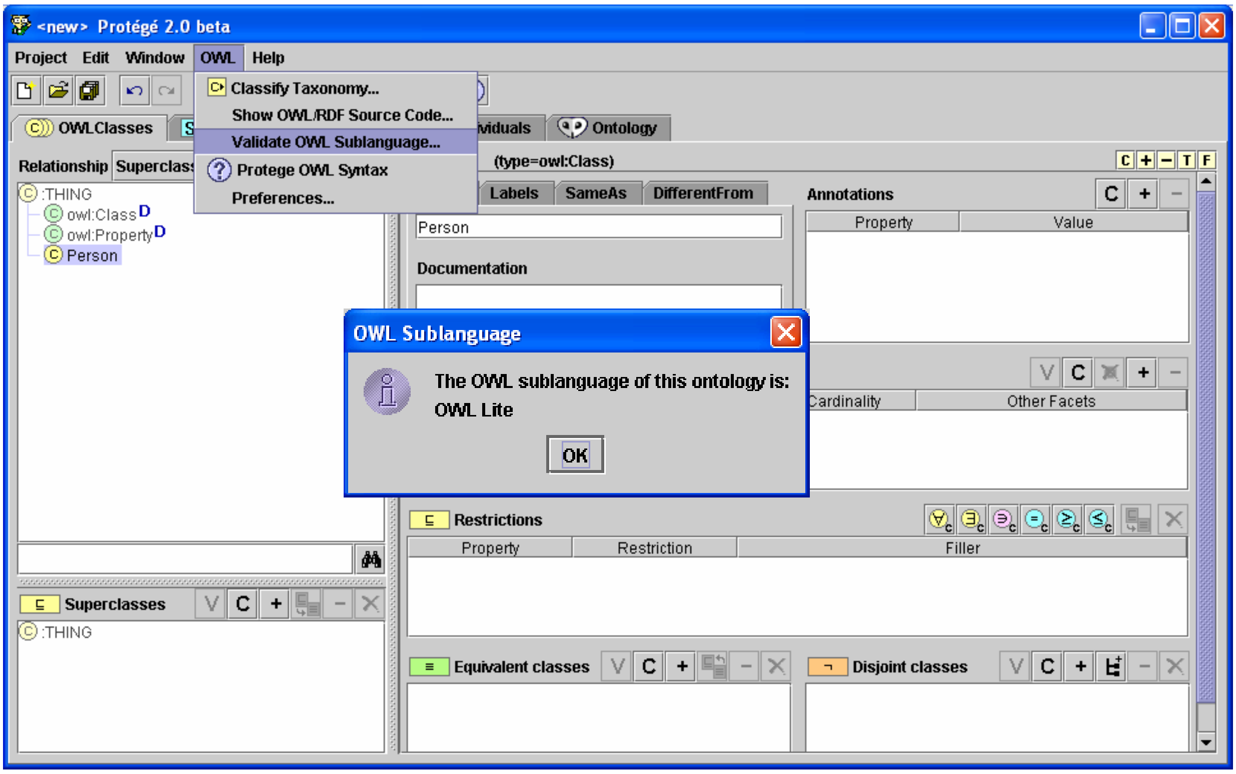
- The OWL Language is divided into three dialects
- OWL Lite:
  - Classification hierarchy
  - Simple restrictions
- OWL DL:
  - Maximal expressiveness while maintaining tractability
  - Standard formalization
- OWL Full:
  - Very high expressiveness (e.g. metaclasses, classes as values)
  - All syntactic freedom of RDF (self-modifying)
  - Loosing tractability (reasoning algorithms become inefficient)

→ **Protégé supports OWL DL and parts of OWL Full**

Protégé / OWL / Species Validator



- The Species Validator determines whether an ontology uses constructs from OWL Lite, OWL DL or OWL Full

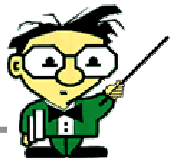


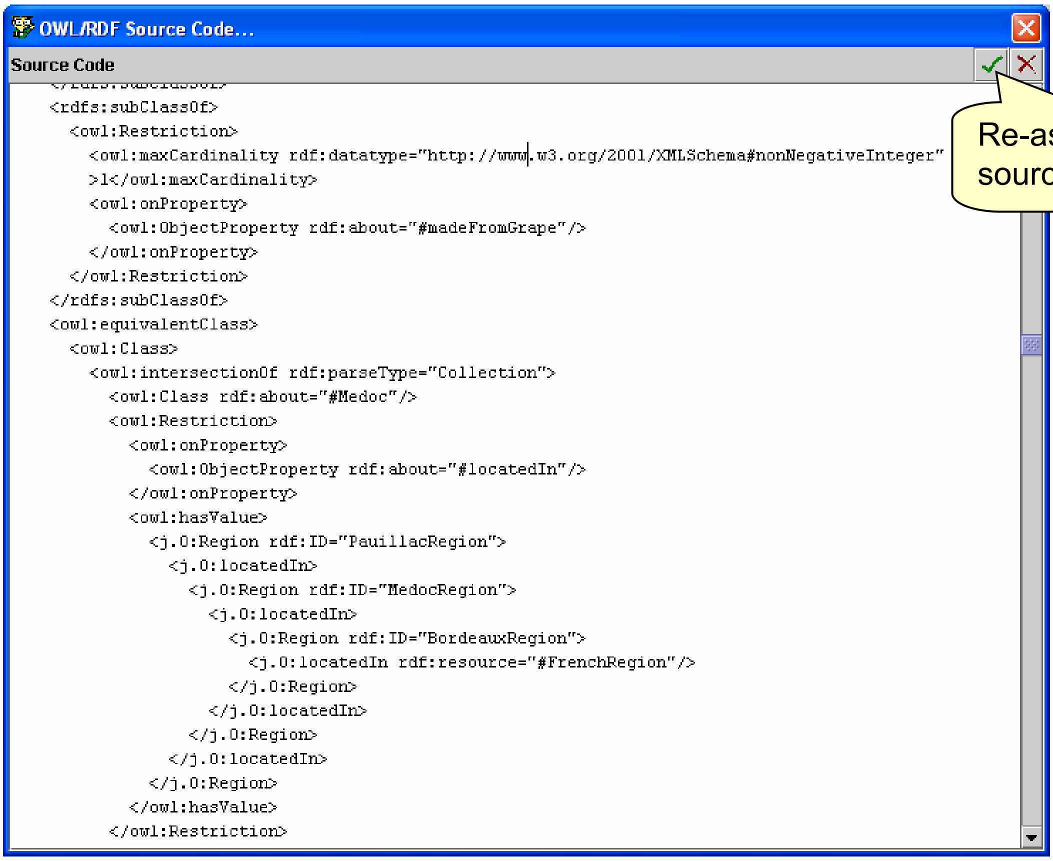
OWL Sublanguage

The OWL sublanguage of this ontology is:
OWL Lite

OK

Protégé / OWL / Source Code Viewer



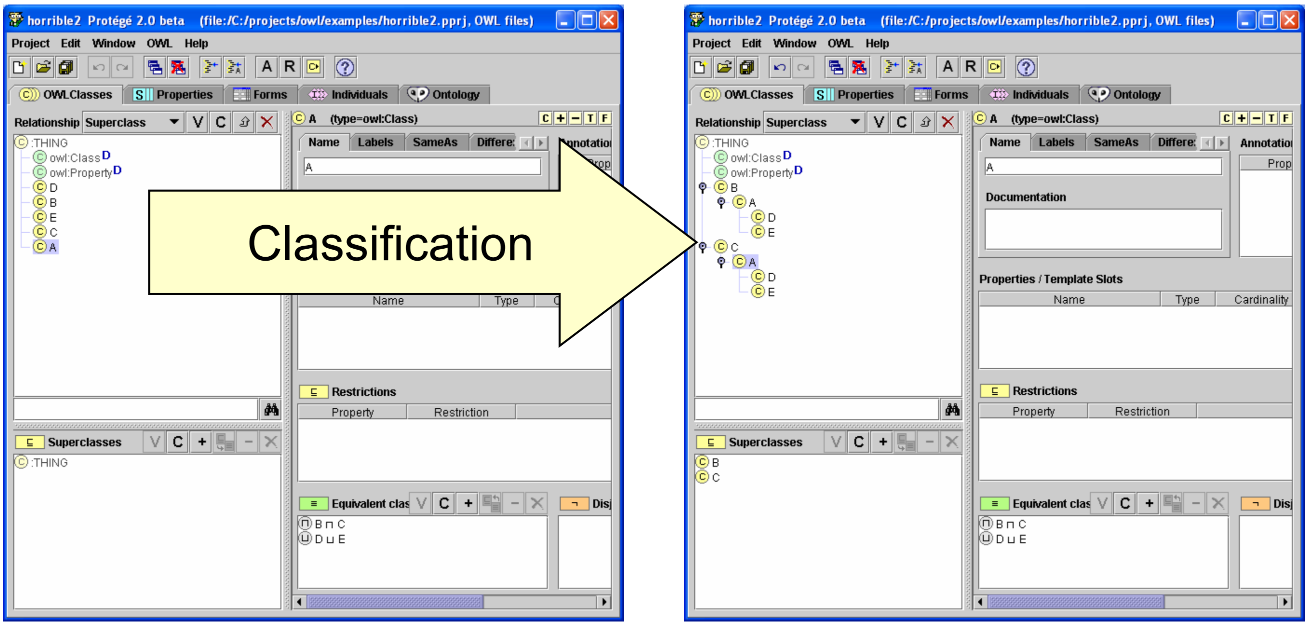


```
</rdfs:subClassOf>
<rdfs:subClassOf>
  <owl:Restriction>
    <owl:maxCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger"
    >1</owl:maxCardinality>
    <owl:onProperty>
      <owl:ObjectProperty rdf:about="#madeFromGrape"/>
    </owl:onProperty>
  </owl:Restriction>
</rdfs:subClassOf>
<owl:equivalentClass>
  <owl:Class>
    <owl:intersectionOf rdf:parseType="Collection">
      <owl:Class rdf:about="#Medoc"/>
      <owl:Restriction>
        <owl:onProperty>
          <owl:ObjectProperty rdf:about="#locatedIn"/>
        </owl:onProperty>
        <owl:hasValue>
          <j.0:Region rdf:ID="PauillacRegion">
            <j.0:locatedIn>
              <j.0:Region rdf:ID="MedocRegion">
                <j.0:locatedIn>
                  <j.0:Region rdf:ID="BordeauxRegion">
                    <j.0:locatedIn rdf:resource="#FrenchRegion"/>
                  </j.0:Region>
                </j.0:locatedIn>
              </j.0:Region>
            </j.0:locatedIn>
          </j.0:Region>
        </owl:hasValue>
      </owl:Restriction>
```

Re-assign from source code

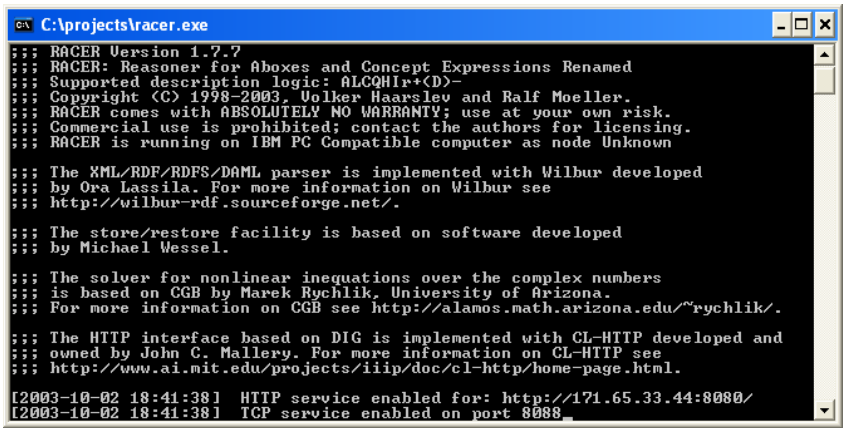
Protégé / OWL / Classification

- OWL’s background of description logics allows to use DL reasoners
- Classification is a reasoner that computes the subsumption relationships (inheritance) between classes based on their logical definition: Result is a new class hierarchy of the existing classes



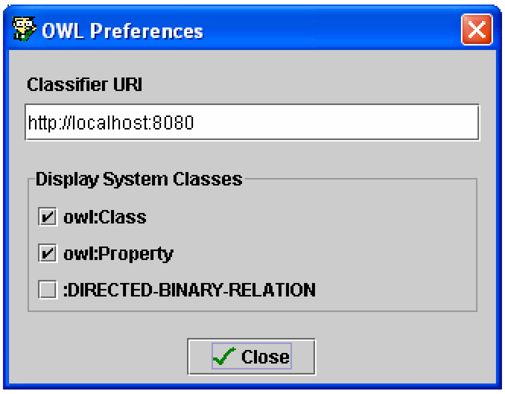
Protégé / OWL / Classification / Getting Started

- Protégé allows the user to plug in external classifiers
- We use the DIG interface to access them (e.g. Racer)
- Classifier must execute as a server process
  - Download Racer
  - Execute it together with Protégé
  - If necessary, configure the classifier’s URI (required for Mac users)



C:\projects\racer.exe

```
;;; RACER Version 1.7.7
;;; RACER: Reasoner for Aboxes and Concept Expressions Renamed
;;; Supported description logic: ALCQHIr+(D)-
;;; Copyright (C) 1998-2003, Volker Haarslev and Ralf Moeller.
;;; RACER comes with ABSOLUTELY NO WARRANTY; use at your own risk.
;;; Commercial use is prohibited; contact the authors for licensing.
;;; RACER is running on IBM PC Compatible computer as node Unknown

;;; The XML/RDF/RDFS/DAML parser is implemented with Wilbur developed
;;; by Ora Lassila. For more information on Wilbur see
;;; http://wilbur-rdf.sourceforge.net/.

;;; The store/restore facility is based on software developed
;;; by Michael Wessel.

;;; The solver for nonlinear inequations over the complex numbers
;;; is based on CGB by Marek Rychlik, University of Arizona.
;;; For more information on CGB see http://alamos.math.arizona.edu/~rychlik/.

;;; The HTTP interface based on DIG is implemented with CL-HTTP developed and
;;; owned by John C. Mallery. For more information on CL-HTTP see
;;; http://www.ai.mit.edu/projects/iiip/doc/cl-http/home-page.html.

[2003-10-02 18:41:38]  HTTP service enabled for: http://171.65.33.44:8080/
[2003-10-02 18:41:38]  TCP service enabled on port 8088
```



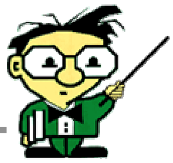
OWL Preferences

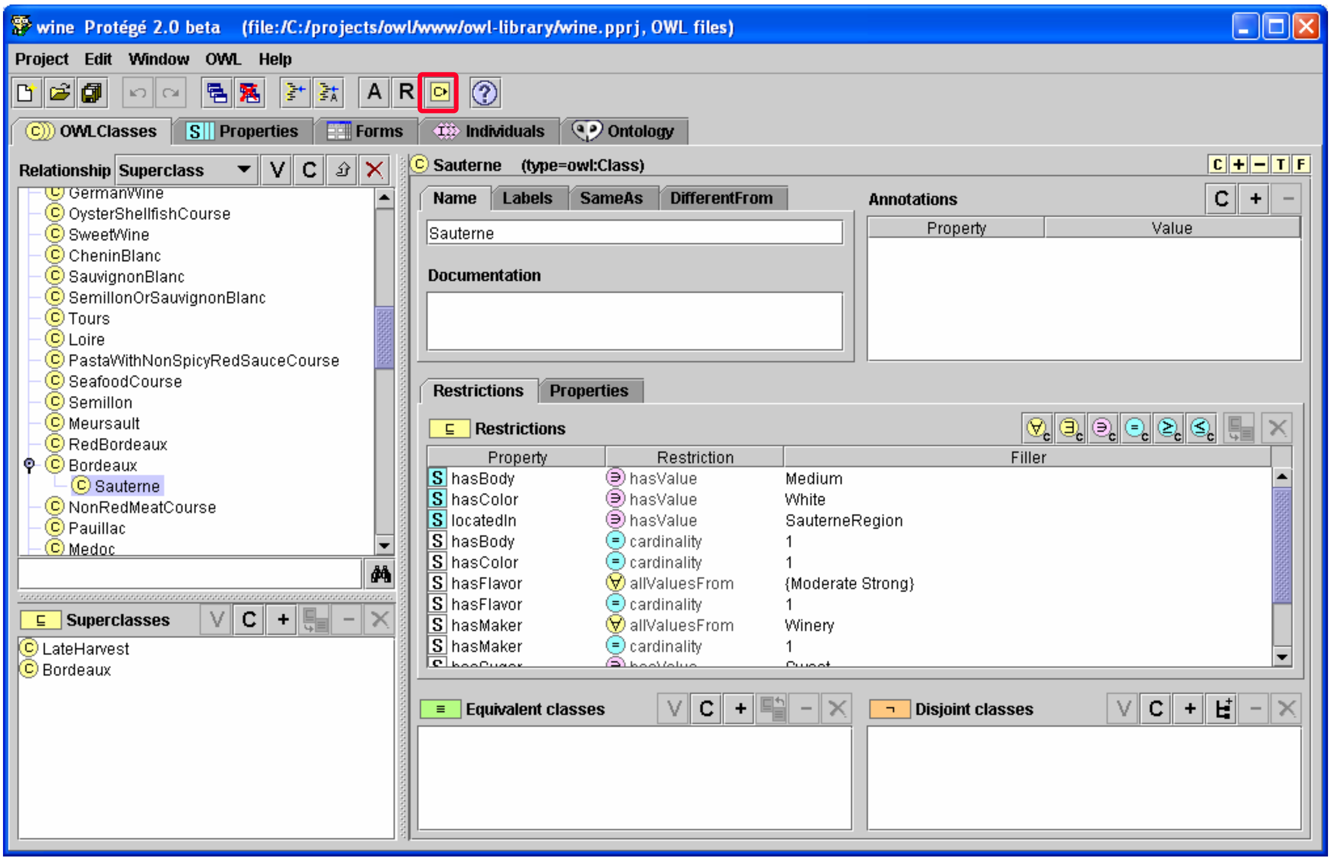
Classifier URI: http://localhost:8080

Display System Classes

- ☑ owl:Class
- ☑ owl:Property
- □ :DIRECTED-BINARY-RELATION

Protégé / OWL / Classification / Example





wine Protégé 2.0 beta (file:/C:/projects/owl/www/owl-library/wine.pprj, OWL files)

GermanWine
OysterShellfishCourse
SweetWine
CheninBlanc
SauvignonBlanc
SemillonOrSauvignonBlanc
Tours
Loire
PastaWithNonSpicyRedSauceCourse
SeafoodCourse
Semillon
Meursault
RedBordeaux
Bordeaux
Sauterne
NonRedMeatCourse
Pauillac
Medoc

Superclasses

LateHarvest
Bordeaux

Sauterne (type=owl:Class)

Sauterne

Documentation

Property	Value

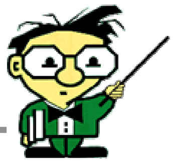
Restrictions

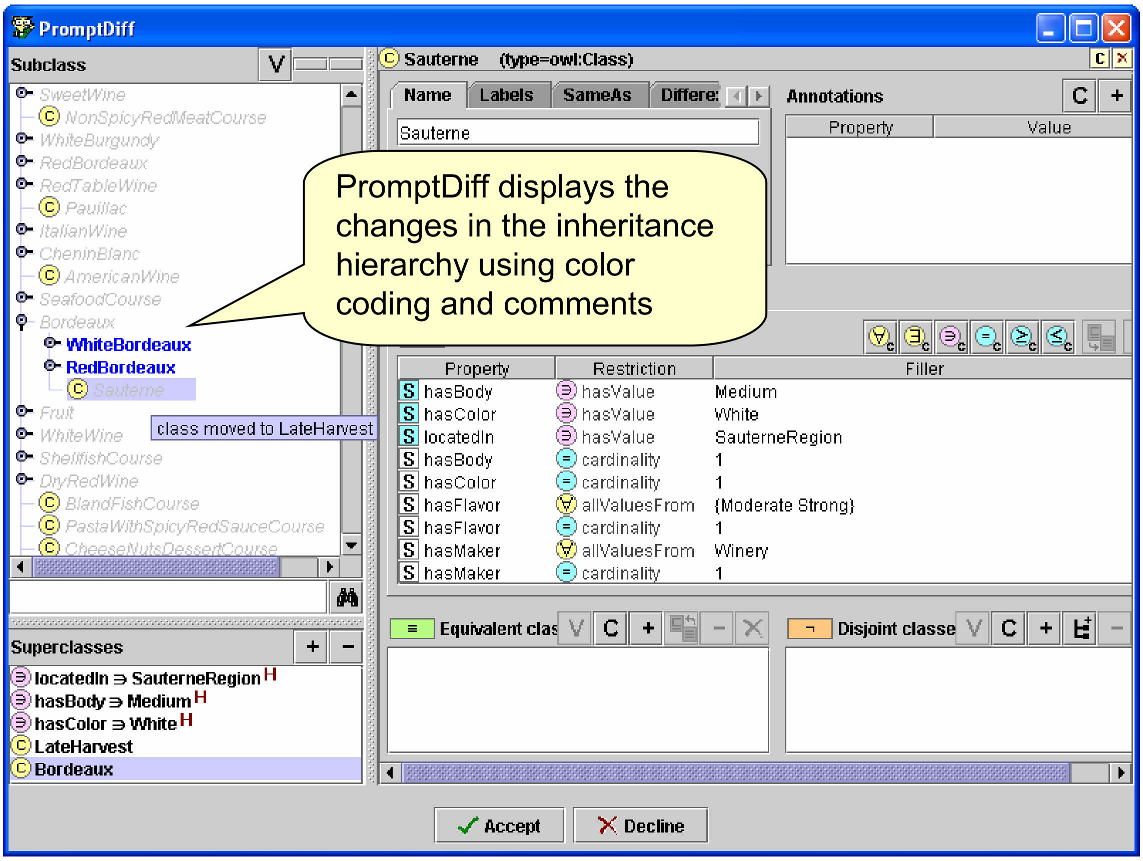
Property	Restriction	Filler
hasBody	hasValue	Medium
hasColor	hasValue	White
locatedIn	hasValue	SauterneRegion
hasBody	cardinality	1
hasColor	cardinality	1
hasFlavor	allValuesFrom	{Moderate Strong}
hasFlavor	cardinality	1
hasMaker	allValuesFrom	Winery
hasMaker	cardinality	1
hasSugar	hasValue	Sweet

Equivalent classes

Disjoint classes

Protégé / OWL / Classification / Example / Prompt





PromptDiff

SweetWine
NonSpicyRedMeatCourse
WhiteBurgundy
RedBordeaux
RedTableWine
Pauillac
ItalianWine
CheninBlanc
AmericanWine
SeafoodCourse
Bordeaux
WhiteBordeaux
RedBordeaux
Sauterne
Fruit
WhiteWine
ShellfishCourse
DryRedWine
BlandFishCourse
PastaWithSpicyRedSauceCourse
CheeseNutsDessertCourse

class moved to LateHarvest

PromptDiff displays the changes in the inheritance hierarchy using color coding and comments

Superclasses

locatedIn ∋ SauterneRegion H
hasBody ∋ Medium H
hasColor ∋ White H
LateHarvest
Bordeaux

Sauterne (type=owl:Class)

Sauterne

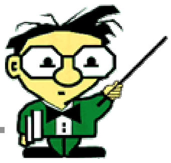
Property	Value

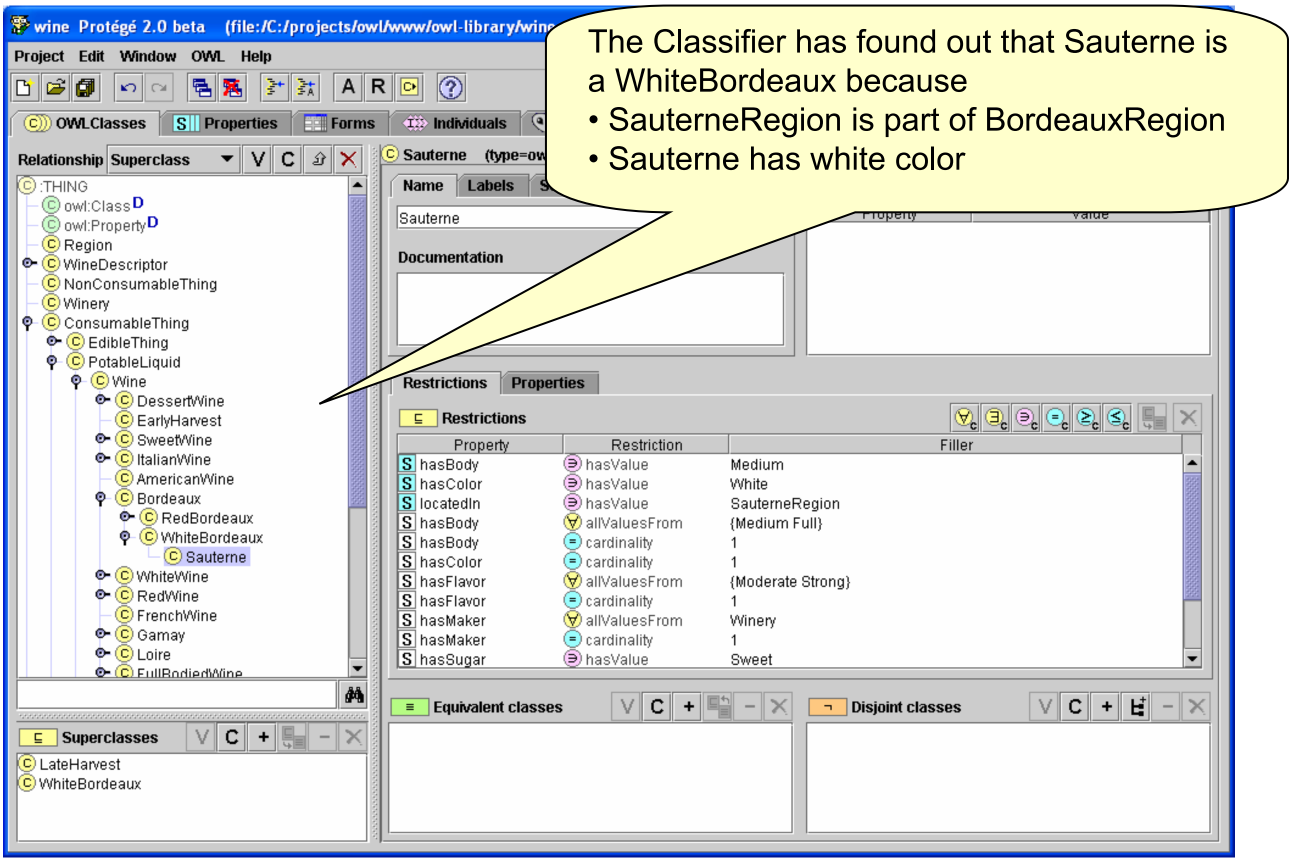
Property	Restriction	Filler
hasBody	hasValue	Medium
hasColor	hasValue	White
locatedIn	hasValue	SauterneRegion
hasBody	cardinality	1
hasColor	cardinality	1
hasFlavor	allValuesFrom	{Moderate Strong}
hasFlavor	cardinality	1
hasMaker	allValuesFrom	Winery
hasMaker	cardinality	1

Equivalent clas

Disjoint classe

Protégé / OWL / Classification / Example / Result





wine Protégé 2.0 beta (file:/C:/projects/owl/www/owl-library/wine

Project Edit Window OWL Help

OWLClasses | Properties | Forms | Individuals

Relationship Superclass

- :THING
  - owl:Class D
  - owl:Property D
  - Region
  - WineDescriptor
  - NonConsumableThing
  - Winery
  - ConsumableThing
    - EdibleThing
    - PotableLiquid
      - Wine
        - DessertWine
        - EarlyHarvest
        - SweetWine
        - ItalianWine
        - AmericanWine
        - Bordeaux
          - RedBordeaux
          - WhiteBordeaux
            - Sauterne
        - WhiteWine
        - RedWine
        - FrenchWine
        - Gamay
        - Loire
        - FullBodiedWine

Superclasses

- LateHarvest
- WhiteBordeaux

Sauterne (type=ow

Name | Labels

Name: Sauterne

Documentation

Property | Value

Restrictions | Properties

Restrictions

Property	Restriction	Filler
hasBody	hasValue	Medium
hasColor	hasValue	White
locatedIn	hasValue	SauterneRegion
hasBody	allValuesFrom	{Medium Full}
hasBody	cardinality	1
hasColor	cardinality	1
hasFlavor	allValuesFrom	{Moderate Strong}
hasFlavor	cardinality	1
hasMaker	allValuesFrom	Winery
hasMaker	cardinality	1
hasSugar	hasValue	Sweet

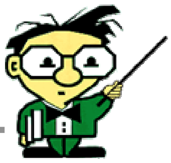
Equivalent classes

Disjoint classes

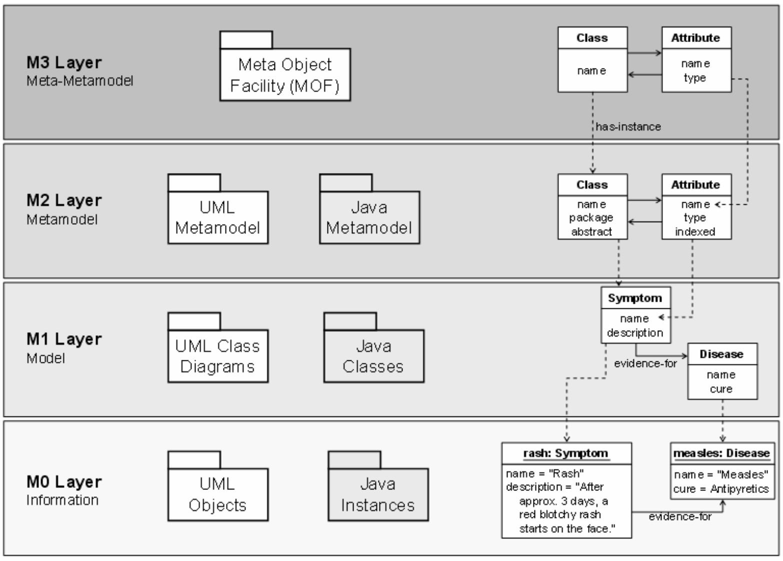
The Classifier has found out that Sauterne is a WhiteBordeaux because

- SauterneRegion is part of BordeauxRegion
- Sauterne has white color

Protégé / OWL / Metaclasses



- Protégé classes and properties are instances of metaclasses
- Metaclasses thus defines the characteristics of other classes
- Traditional object-oriented systems have a similar structure:



M3 Layer Meta-Metamodel: Meta Object Facility (MOF); Class (name) — Attribute (name, type)

has-instance

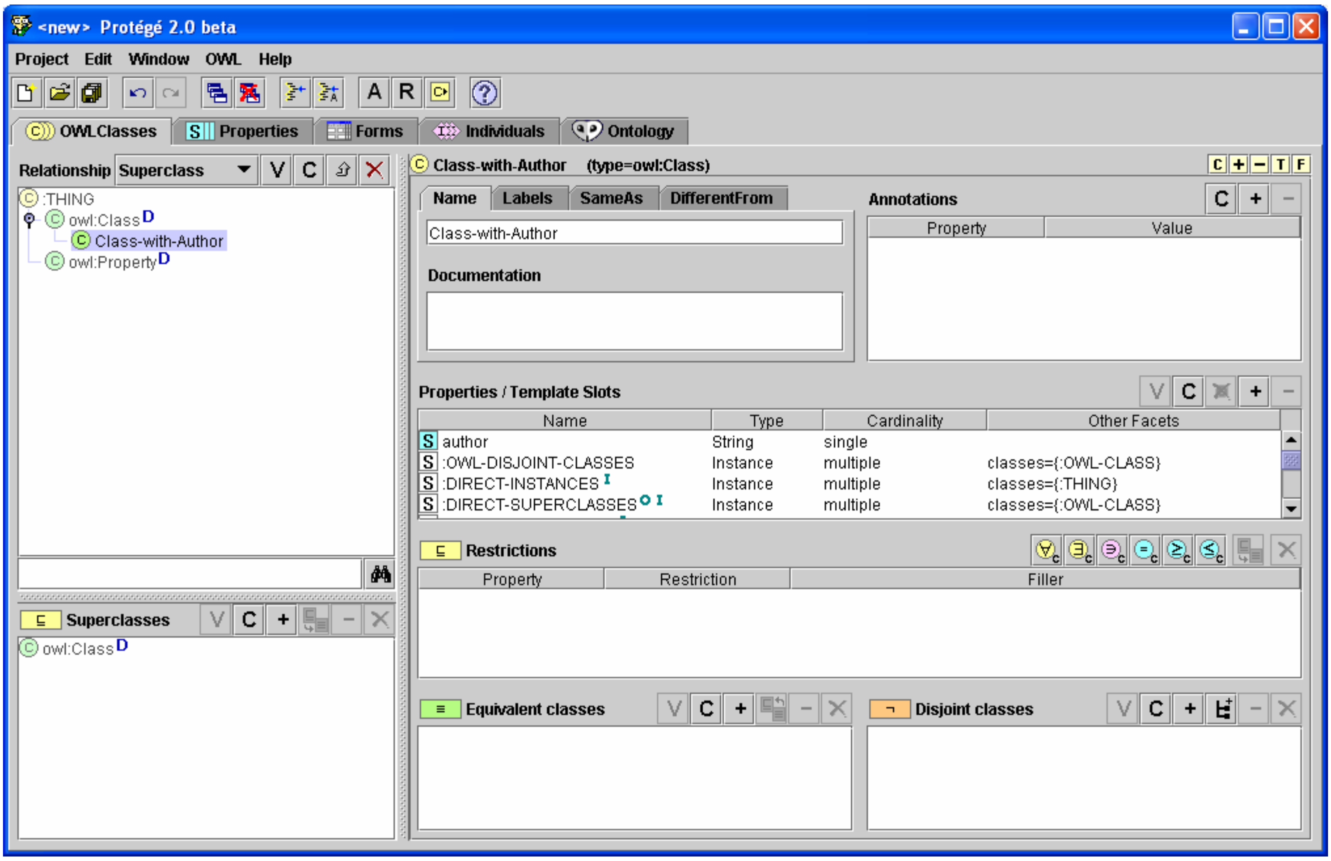
M2 Layer Metamodel: UML Metamodel; Java Metamodel; Class (name, package, abstract) — Attribute (name, type, indexed)

M1 Layer Model: UML Class Diagrams; Java Classes; Symptom (name, description) — evidence-for → Disease (name, cure)

M0 Layer Information: UML Objects; Java Instances; rash: Symptom (name = "Rash", description = "After approx. 3 days, a red blotchy rash starts on the face.") — evidence-for → measles: Disease (name = "Measles", cure = Antipyretics)

- Protégé provides its own metamodel as a Protégé ontology
- The metamodel can be extended by plugins and by users

Protégé / OWL / Metaclasses / Creating Metaclasses



:THING
- owl:Class D
  - Class-with-Author
- owl:Property D

Class-with-Author (type=owl:Class)

Name: Class-with-Author

Documentation

Annotations

Property	Value

Properties / Template Slots

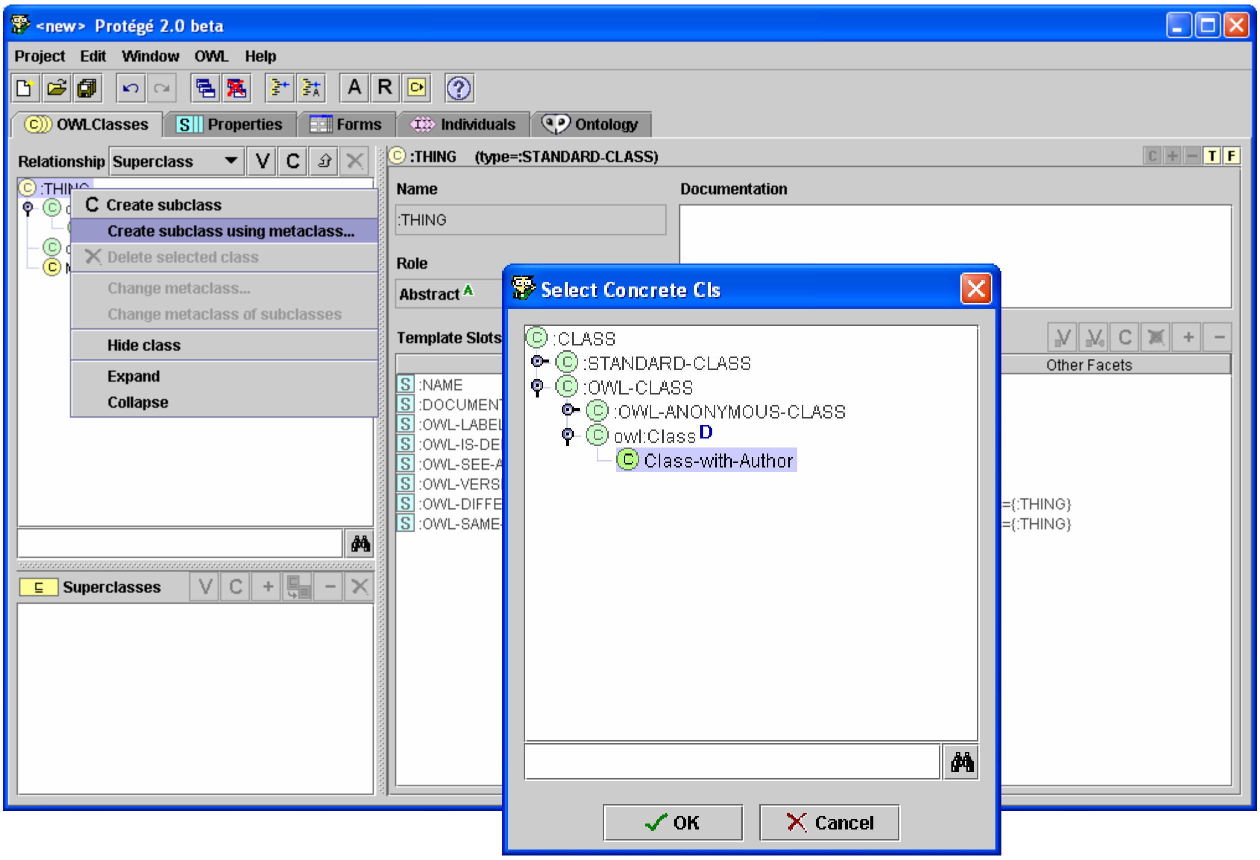
Name	Type	Cardinality	Other Facets
S author	String	single	
S :OWL-DISJOINT-CLASSES	Instance	multiple	classes={:OWL-CLASS}
S :DIRECT-INSTANCES I	Instance	multiple	classes={:THING}
S :DIRECT-SUPERCLASSES O I	Instance	multiple	classes={:OWL-CLASS}

Restrictions

Property	Restriction	Filler

Superclasses: owl:Class D

Protégé / OWL / Metaclasses / Creating Classes



:THING (type=:STANDARD-CLASS)

Name: :THING

Documentation

Role: Abstract A

Template Slots

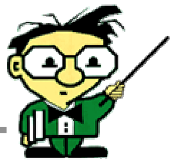
S :NAME
S :DOCUMENT
S :OWL-LABEL
S :OWL-IS-DE
S :OWL-SEE-A
S :OWL-VERS
S :OWL-DIFFE ={:THING}
S :OWL-SAME- ={:THING}

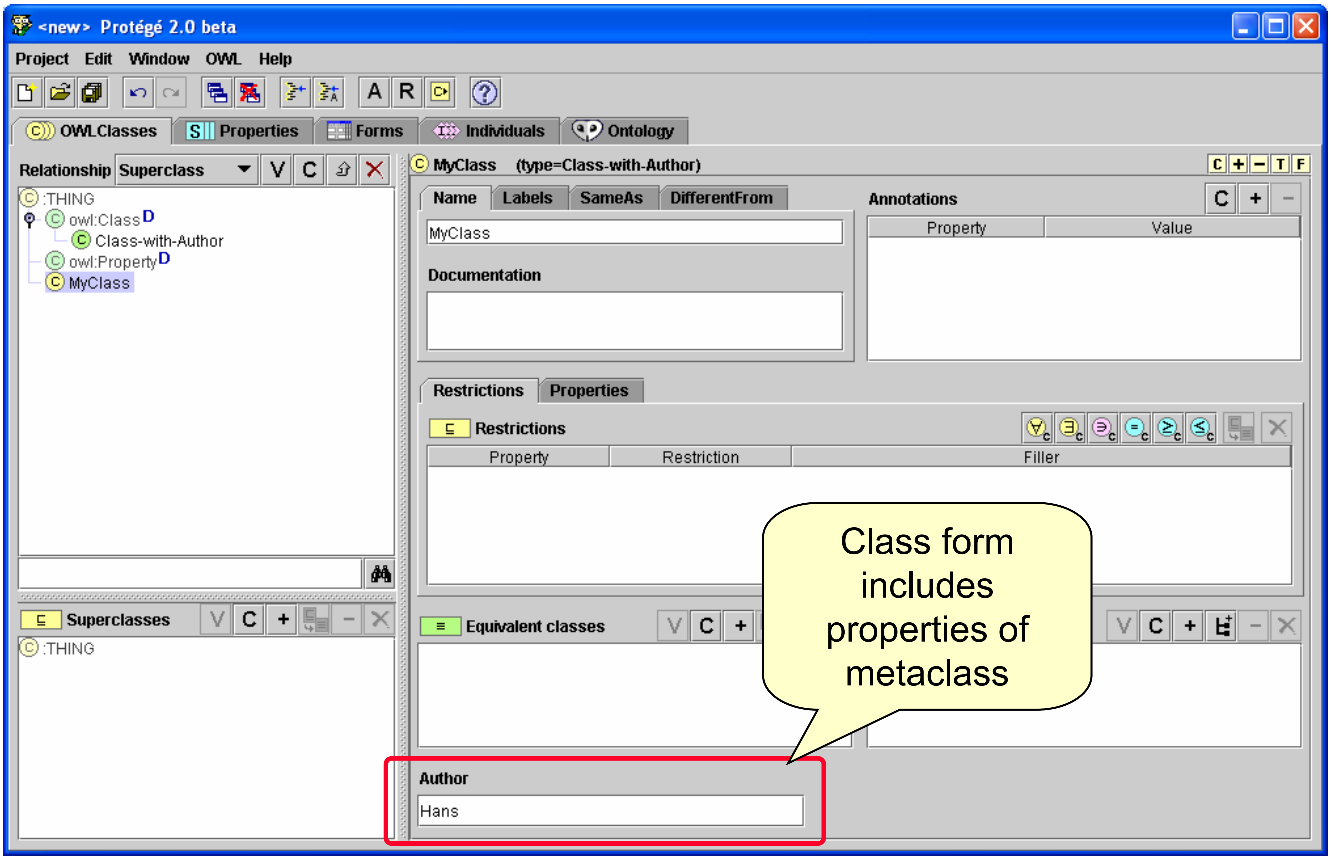
Other Facets

Select Concrete Cls:

- :CLASS
  - :STANDARD-CLASS
  - :OWL-CLASS
    - :OWL-ANONYMOUS-CLASS
    - owl:Class D
      - Class-with-Author

Protégé / OWL / Metaclasses / Creating Classes



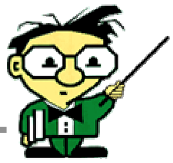


Name: MyClass

Author: Hans

Class form includes properties of metaclass

Protégé / OWL / Metaclasses / Usage

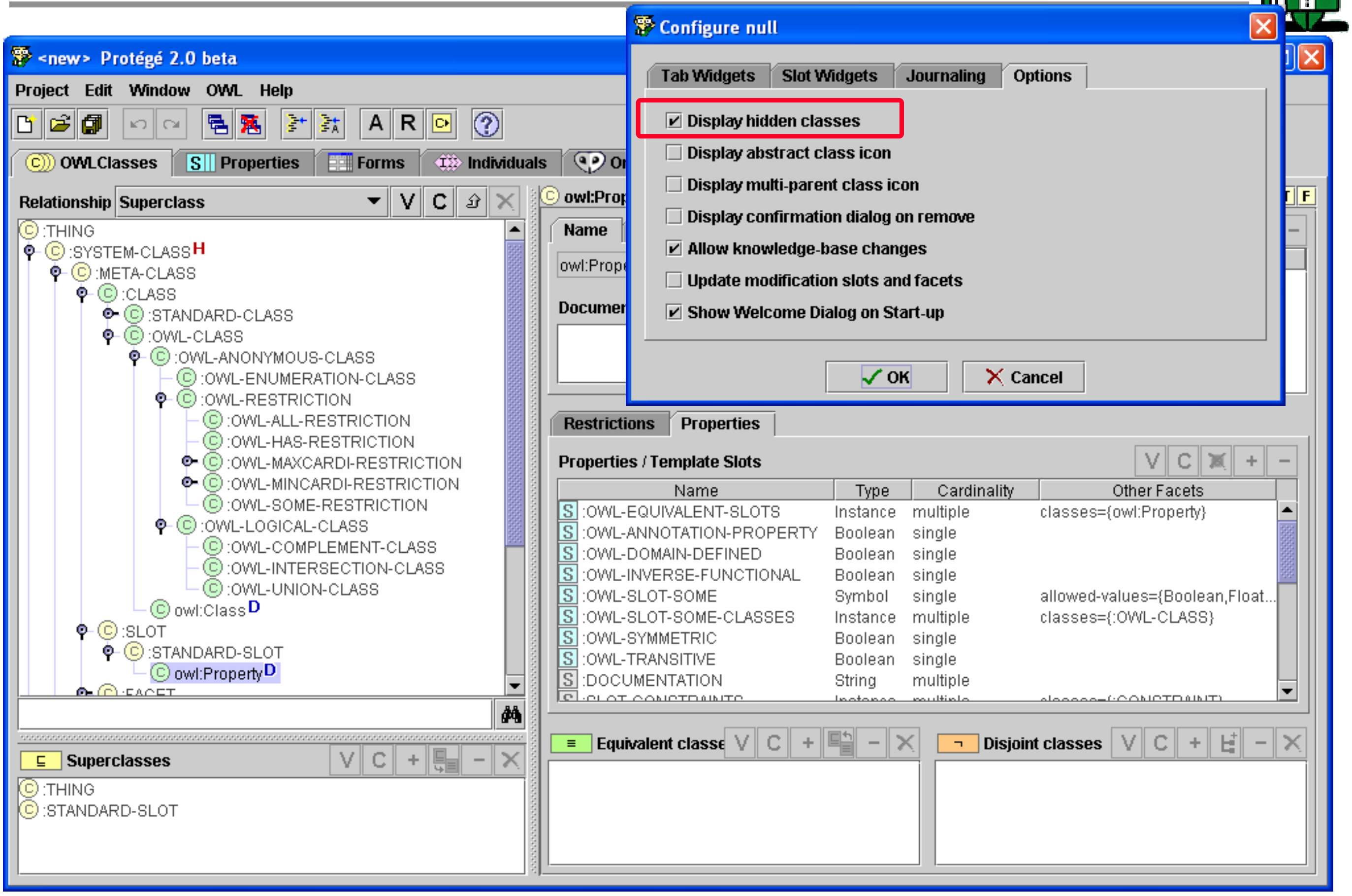


- Use of metaclasses means your ontology is in OWL Full
  → **Limited reasoning support due to intractability**

- Alternative: Annotation properties
  - Can be assigned to any ontology resource
  - Free choice of properties at any time

- But...
  - Metaclasses add widgets to the class editing form
  - Make explicit which values should be entered
  - Guide the knowledge-acquisition process

Protégé / OWL / Metaclasses / System metaclasses



- ☑ Display hidden classes
- □ Display abstract class icon
- □ Display multi-parent class icon
- □ Display confirmation dialog on remove
- ☑ Allow knowledge-base changes
- □ Update modification slots and facets
- ☑ Show Welcome Dialog on Start-up

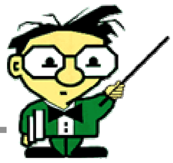
```
:THING
  :SYSTEM-CLASS H
    :META-CLASS
      :CLASS
        :STANDARD-CLASS
        :OWL-CLASS
          :OWL-ANONYMOUS-CLASS
            :OWL-ENUMERATION-CLASS
            :OWL-RESTRICTION
              :OWL-ALL-RESTRICTION
              :OWL-HAS-RESTRICTION
              :OWL-MAXCARDI-RESTRICTION
              :OWL-MINCARDI-RESTRICTION
              :OWL-SOME-RESTRICTION
            :OWL-LOGICAL-CLASS
              :OWL-COMPLEMENT-CLASS
              :OWL-INTERSECTION-CLASS
              :OWL-UNION-CLASS
          owl:Class D
      :SLOT
        :STANDARD-SLOT
          owl:Property D
      :FACET
```

Name	Type	Cardinality	Other Facets
:OWL-EQUIVALENT-SLOTS	Instance	multiple	classes={owl:Property}
:OWL-ANNOTATION-PROPERTY	Boolean	single	
:OWL-DOMAIN-DEFINED	Boolean	single	
:OWL-INVERSE-FUNCTIONAL	Boolean	single	
:OWL-SLOT-SOME	Symbol	single	allowed-values={Boolean,Float...
:OWL-SLOT-SOME-CLASSES	Instance	multiple	classes={:OWL-CLASS}
:OWL-SYMMETRIC	Boolean	single	
:OWL-TRANSITIVE	Boolean	single	
:DOCUMENTATION	String	multiple	
:SLOT-CONSTRAINTS	Instance	multiple	classes={:CONSTRAINT}

Superclasses:

- :THING
- :STANDARD-SLOT

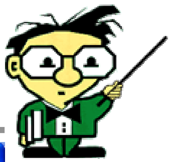
Protégé / OWL / API

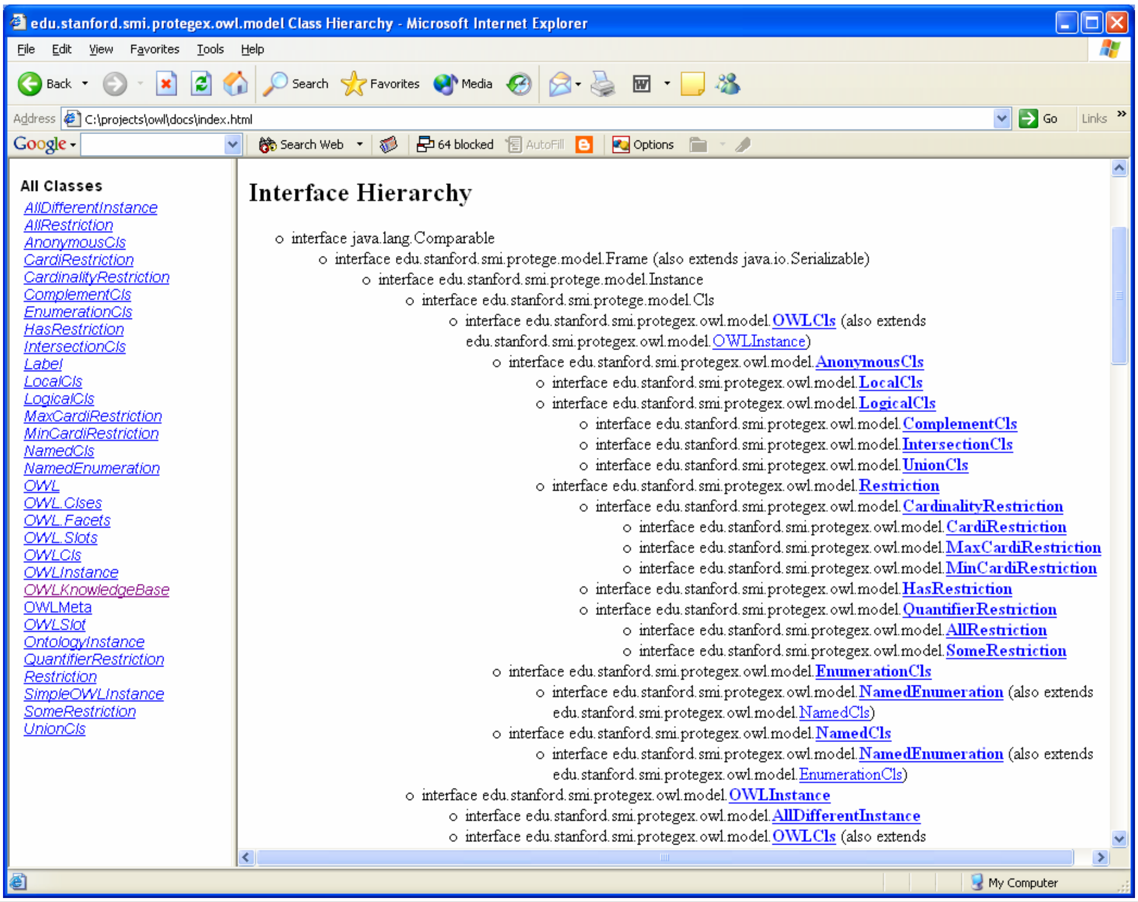


- OWL is a relatively new language
- Few tools and libraries exist to parse and manipulate OWL files

- Jena API (HP Labs, Bristol)
  - RDF parser library
  - Focus on RDF syntax trees
  - Large user community
  - Insufficient event support
- OWL API (Manchester, Karlsruhe)
  - Pure OWL API
  - Focus on OWL Abstract syntax
  - Very new
- Protégé/OWL API
  - Optimized for component development for Protégé
  - Uses Jena API and OWL API for special services

Protégé / OWL / API / Class Overview





edu.stanford.smi.protegex.owl.model Class Hierarchy - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address C:\projects\owl\docs\index.html Go Links
Google Search Web 64 blocked AutoFill Options
My Computer

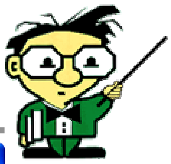
All Classes

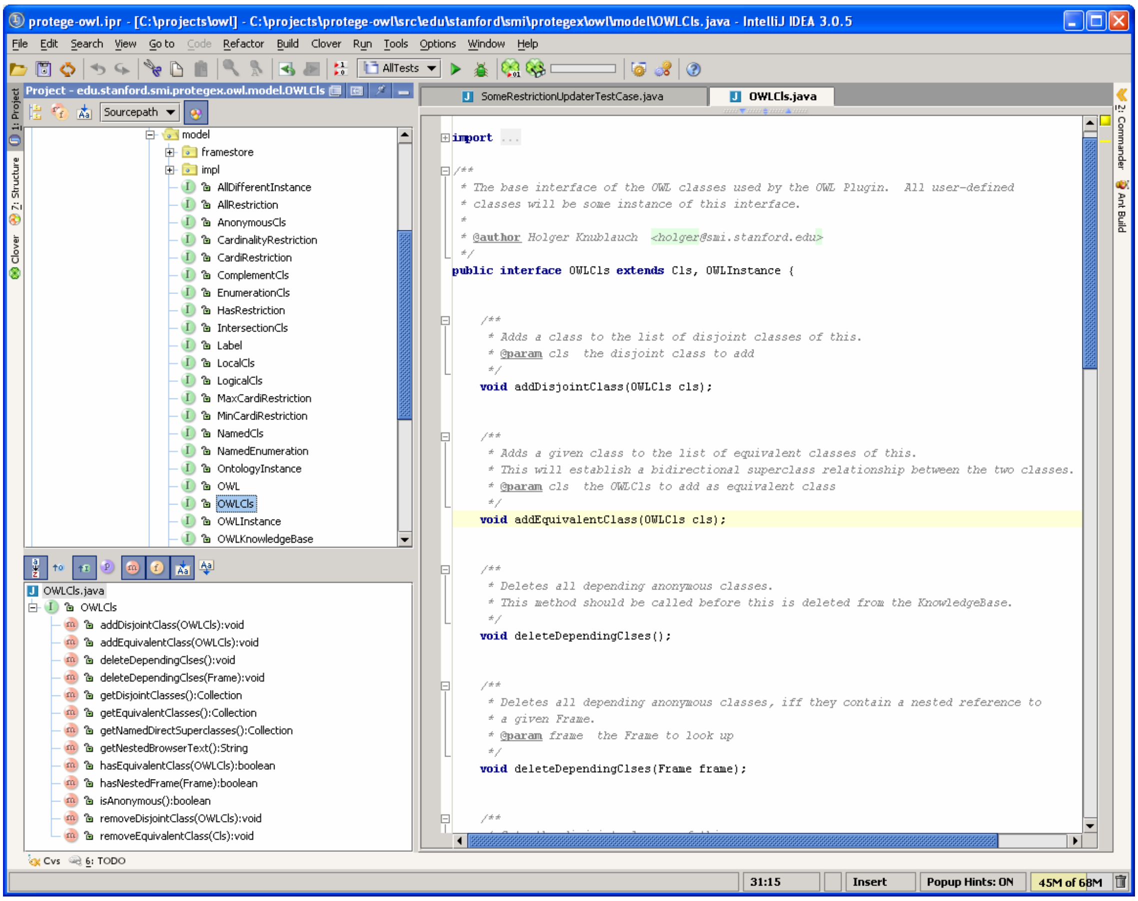
AllDifferentInstance
AllRestriction
AnonymousCls
CardiRestriction
CardinalityRestriction
ComplementCls
EnumerationCls
HasRestriction
IntersectionCls
Label
LocalCls
LogicalCls
MaxCardiRestriction
MinCardiRestriction
NamedCls
NamedEnumeration
OWL
OWL.Clses
OWL.Facets
OWL.Slots
OWLCls
OWLInstance
OWLKnowledgeBase
OWLMeta
OWLSlot
OntologyInstance
QuantifierRestriction
Restriction
SimpleOWLInstance
SomeRestriction
UnionCls

Interface Hierarchy

- interface java.lang.Comparable
  - interface edu.stanford.smi.protege.model.Frame (also extends java.io.Serializable)
    - interface edu.stanford.smi.protege.model.Instance
      - interface edu.stanford.smi.protege.model.Cls
        - interface edu.stanford.smi.protegex.owl.model.OWLCls (also extends edu.stanford.smi.protegex.owl.model.OWLInstance)
          - interface edu.stanford.smi.protegex.owl.model.AnonymousCls
            - interface edu.stanford.smi.protegex.owl.model.LocalCls
            - interface edu.stanford.smi.protegex.owl.model.LogicalCls
              - interface edu.stanford.smi.protegex.owl.model.ComplementCls
              - interface edu.stanford.smi.protegex.owl.model.IntersectionCls
              - interface edu.stanford.smi.protegex.owl.model.UnionCls
            - interface edu.stanford.smi.protegex.owl.model.Restriction
              - interface edu.stanford.smi.protegex.owl.model.CardinalityRestriction
                - interface edu.stanford.smi.protegex.owl.model.CardiRestriction
                - interface edu.stanford.smi.protegex.owl.model.MaxCardiRestriction
                - interface edu.stanford.smi.protegex.owl.model.MinCardiRestriction
              - interface edu.stanford.smi.protegex.owl.model.HasRestriction
              - interface edu.stanford.smi.protegex.owl.model.QuantifierRestriction
                - interface edu.stanford.smi.protegex.owl.model.AllRestriction
                - interface edu.stanford.smi.protegex.owl.model.SomeRestriction
          - interface edu.stanford.smi.protegex.owl.model.EnumerationCls
            - interface edu.stanford.smi.protegex.owl.model.NamedEnumeration (also extends edu.stanford.smi.protegex.owl.model.NamedCls)
          - interface edu.stanford.smi.protegex.owl.model.NamedCls
            - interface edu.stanford.smi.protegex.owl.model.NamedEnumeration (also extends edu.stanford.smi.protegex.owl.model.EnumerationCls)
      - interface edu.stanford.smi.protegex.owl.model.OWLInstance
        - interface edu.stanford.smi.protegex.owl.model.AllDifferentInstance
        - interface edu.stanford.smi.protegex.owl.model.OWLCls (also extends

Protégé / OWL / API





protege-owl.ipr - [C:\projects\owl] - C:\projects\protege-owl\src\edu\stanford\smi\protegex\owl\model\OWLCls.java - IntelliJ IDEA 3.0.5
File Edit Search View Go to Code Refactor Build Clover Run Tools Options Window Help
AllTests
Project - edu.stanford.smi.protegex.owl.model.OWLCls
Sourcepath
1: Project
7: Structure
Clover
SomeRestrictionUpdaterTestCase.java
OWLCls.java
2: Commander
Ant Build
Cvs
6: TODO
31:15
Insert
Popup Hints: ON
45M of 68M

model
- framestore
- impl
- AllDifferentInstance
- AllRestriction
- AnonymousCls
- CardinalityRestriction
- CardiRestriction
- ComplementCls
- EnumerationCls
- HasRestriction
- IntersectionCls
- Label
- LocalCls
- LogicalCls
- MaxCardiRestriction
- MinCardiRestriction
- NamedCls
- NamedEnumeration
- OntologyInstance
- OWL
- OWLCls
- OWLInstance
- OWLKnowledgeBase

OWLCls.java
- OWLCls
  - addDisjointClass(OWLCls):void
  - addEquivalentClass(OWLCls):void
  - deleteDependingClses():void
  - deleteDependingClses(Frame):void
  - getDisjointClasses():Collection
  - getEquivalentClasses():Collection
  - getNamedDirectSuperclasses():Collection
  - getNestedBrowserText():String
  - hasEquivalentClass(OWLCls):boolean
  - hasNestedFrame(Frame):boolean
  - isAnonymous():boolean
  - removeDisjointClass(OWLCls):void
  - removeEquivalentClass(Cls):void

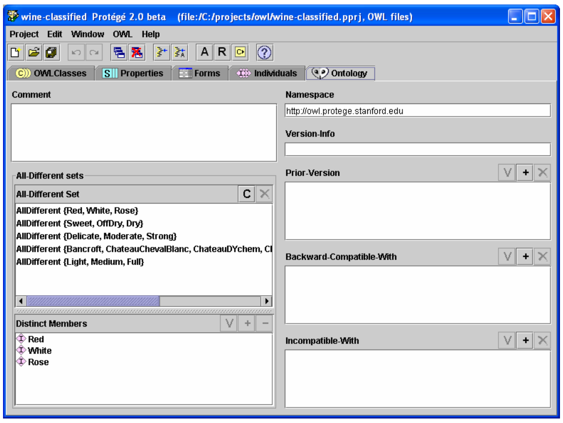
```
import ...

/**
 * The base interface of the OWL classes used by the OWL Plugin.  All user-defined
 * classes will be some instance of this interface.
 *
 * @author Holger Knublauch  <holger@smi.stanford.edu>
 */
public interface OWLCls extends Cls, OWLInstance {


    /**
     * Adds a class to the list of disjoint classes of this.
     * @param cls  the disjoint class to add
     */
    void addDisjointClass(OWLCls cls);


    /**
     * Adds a given class to the list of equivalent classes of this.
     * This will establish a bidirectional superclass relationship between the two classes.
     * @param cls  the OWLCls to add as equivalent class
     */
    void addEquivalentClass(OWLCls cls);


    /**
     * Deletes all depending anonymous classes.
     * This method should be called before this is deleted from the KnowledgeBase.
     */
    void deleteDependingClses();


    /**
     * Deletes all depending anonymous classes, iff they contain a nested reference to
     * a given Frame.
     * @param frame  the Frame to look up
     */
    void deleteDependingClses(Frame frame);


    /**
```

Protégé / OWL / API / Writing Applications

```
OWLKnowledgeBase okb = ...

NamedCls personCls = okb.createNamedCls("Person");
NamedCls aCls = okb.createNamedCls("HappyPerson");
NamedCls bCls = okb.createNamedCls("OtherPerson");
aCls.addDirectSuperclass(personCls);
bCls.addDirectSuperclass(personCls);

OWLSlot slot = okb.createOWLSlot("children");
slot.setValueType(ValueType.INSTANCE);
slot.setAllowedClses(Collections.singleton(personCls));
personCls.addDirectTemplateSlot(slot);

SomeRestriction aRestriction =
        okb.createSomeRestriction(slot, aCls);
personCls.addDirectSuperclass(aRestriction);
```

Protégé / OWL / API / Plugins

- Protégé is an open platform that allows extensions (plugins)
  - Storage Plugins: Load and save ontologies in various formats
  - Tab Plugins: Provide a top-level user interface component
  - Slot Widgets: Customized components for editing property values
- Developing plugins is really simple!



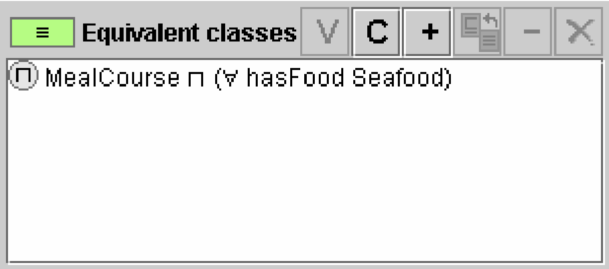
Namespace: http://owl.protege.stanford.edu

Version-Info:

AllDifferent {Red, White, Rose}
AllDifferent {Sweet, OffDry, Dry}
AllDifferent {Delicate, Moderate, Strong}
AllDifferent {Bancroft, ChateauChevalBlanc, ChateauDYchem, Cl
AllDifferent {Light, Medium, Full}

Red
White
Rose

A Tab Plugin (“Ontology”)

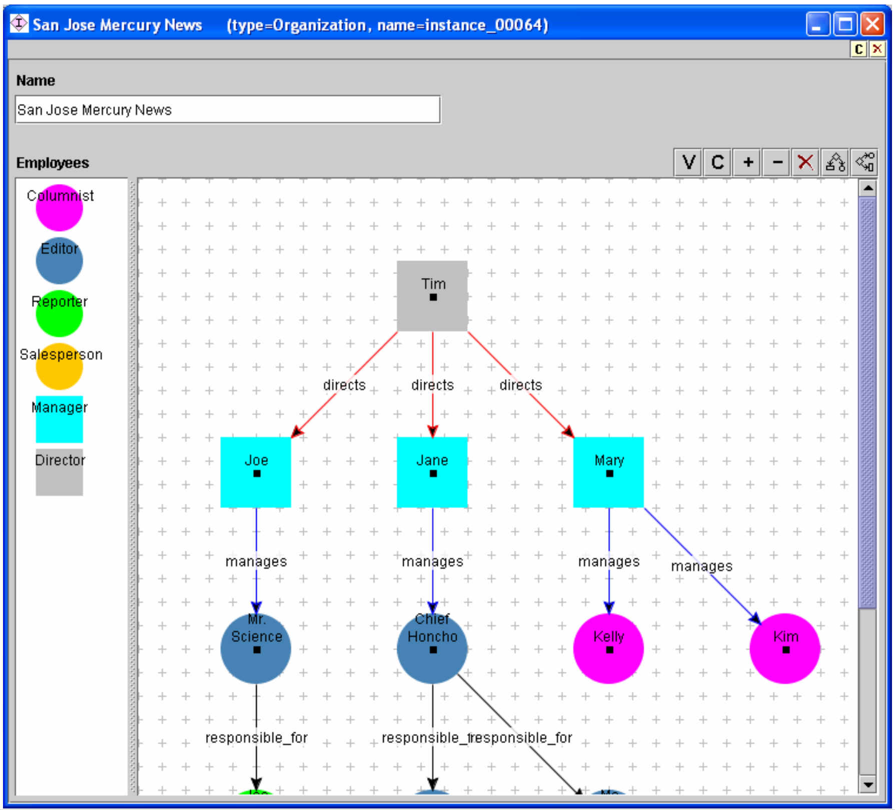


MealCourse ⊓ (∀ hasFood Seafood)

A Slot Widget Plugin

Protégé / OWL / Graph Widget

The GraphWidget can be used to edit instance properties visually



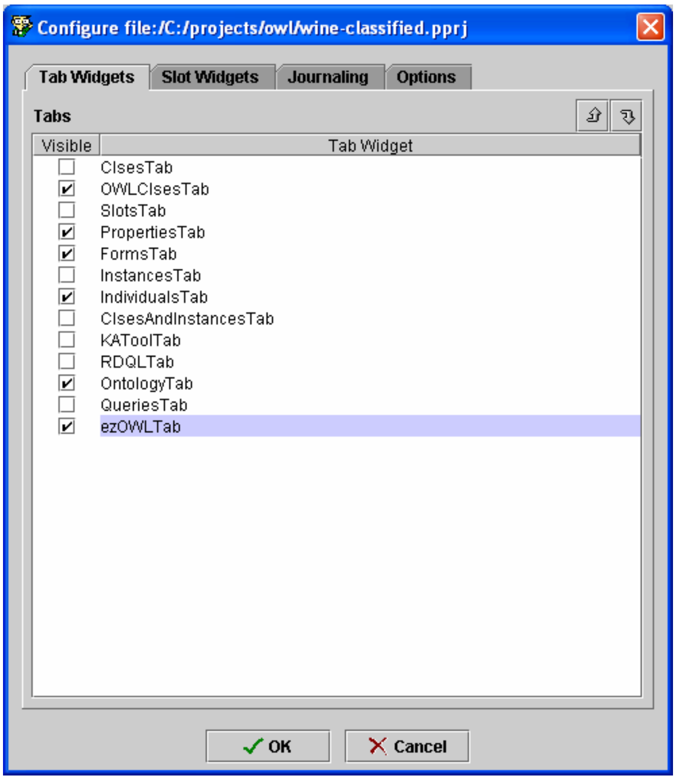
San Jose Mercury News (type=Organization, name=instance_00064)

Name: San Jose Mercury News

Employees

Protégé / OWL / API / Plugins / Installation

- Many Plugins can be downloaded from Protégé web site
- They usually consist of one or more .jar files
- Copy these .jar files into the plugins folder of your Protégé installation
- Tab plugins usually have to be activated before you see them (Project/Configure...)

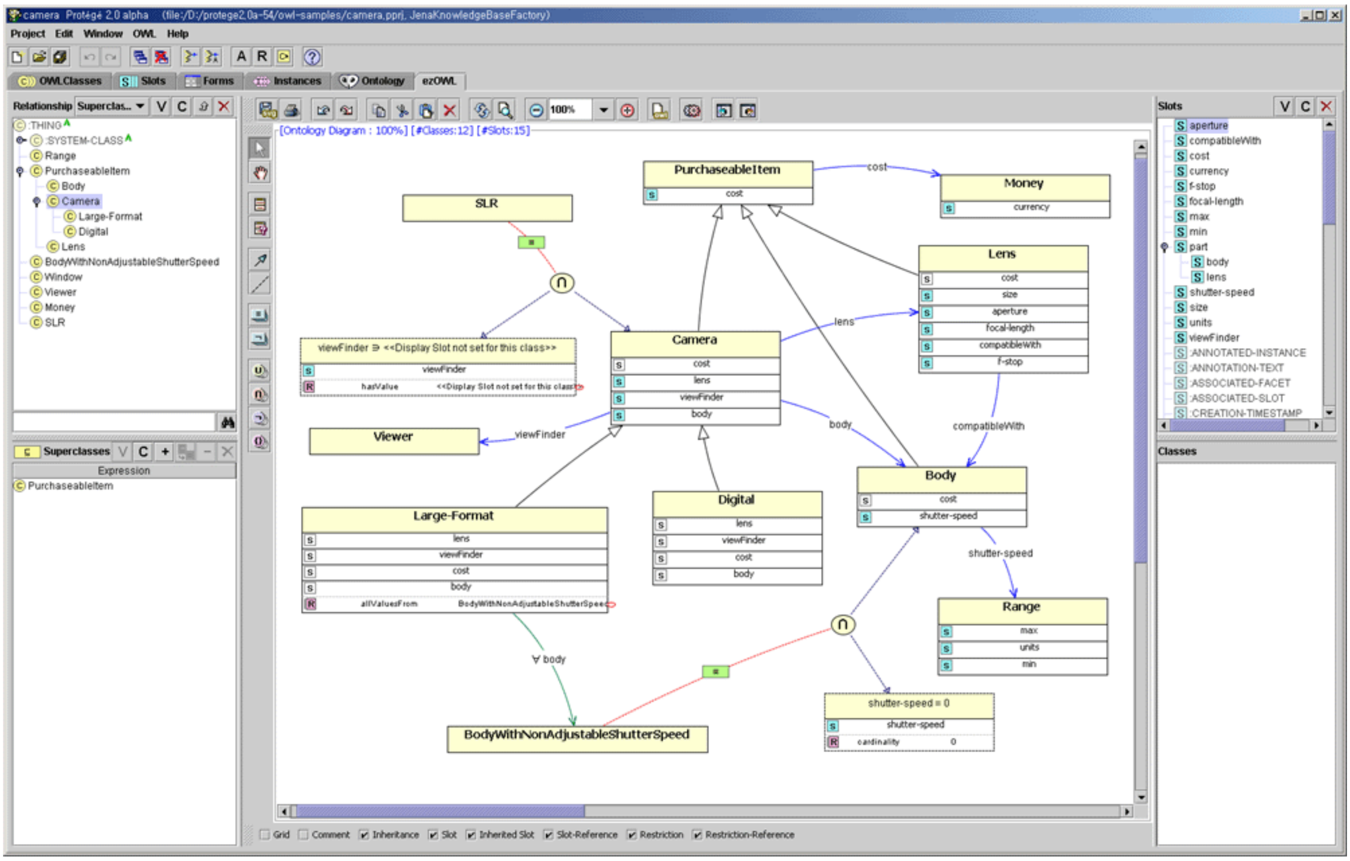


Configure file:/C:/projects/owl/wine-classified.pprj

Tabs

Visible	Tab Widget
□	ClsesTab
☑	OWLClsesTab
□	SlotsTab
☑	PropertiesTab
☑	FormsTab
□	InstancesTab
☑	IndividualsTab
□	ClsesAndInstancesTab
□	KAToolTab
□	RDQLTab
☑	OntologyTab
□	QueriesTab
☑	ezOWLTab

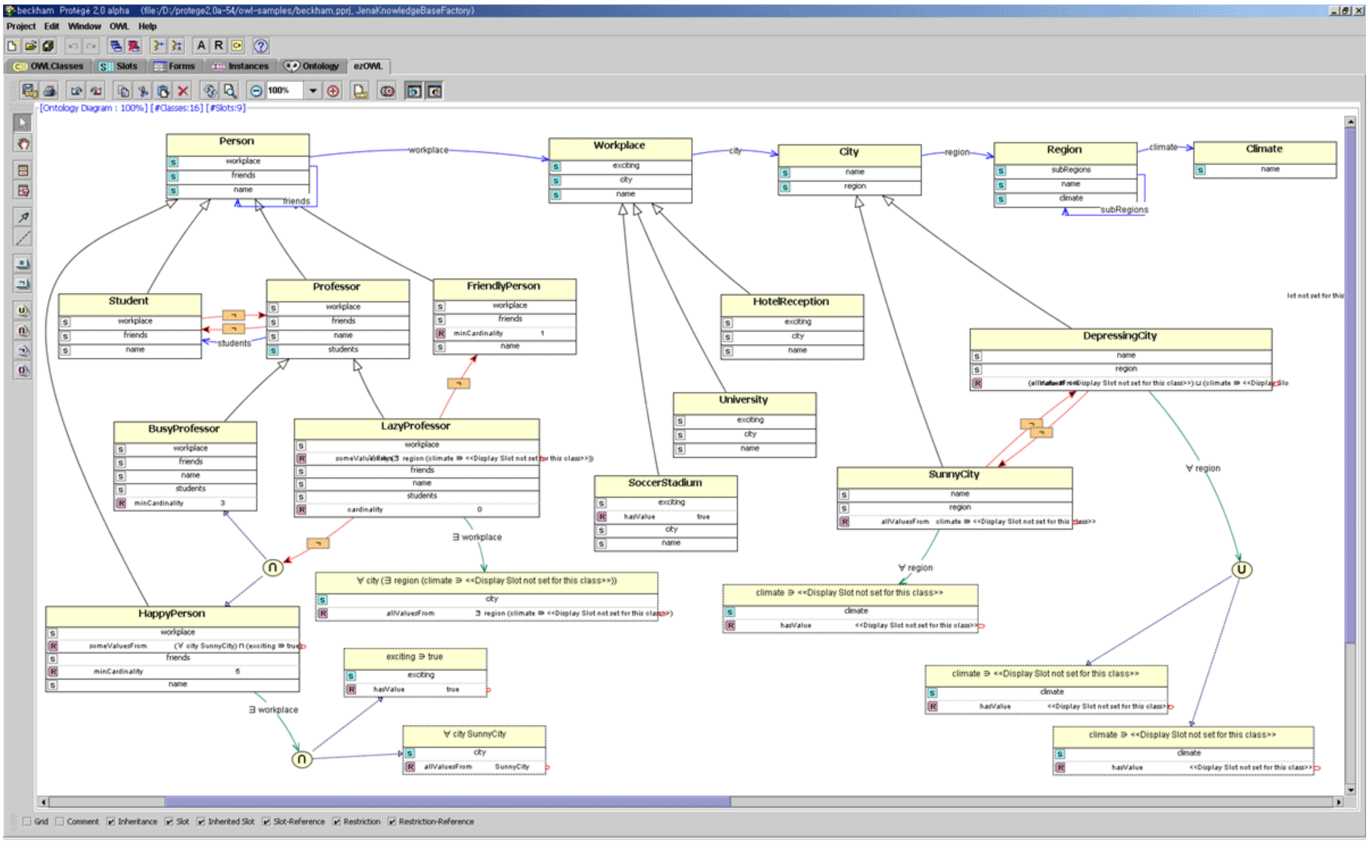
Protégé / OWL / Plugins / ezOWL (1)



camera Protégé 2.0 alpha (file:/D:/protege2.0a-54/owl-samples/camera.pprj, JenaKnowledgeBaseFactory)

□ Grid □ Comment ☑ Inheritance ☑ Slot ☑ Inherited Slot ☑ Slot-Reference ☑ Restriction ☑ Restriction-Reference

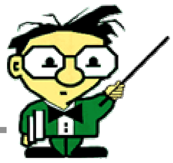
Protégé / OWL / Plugins / ezOWL (2)

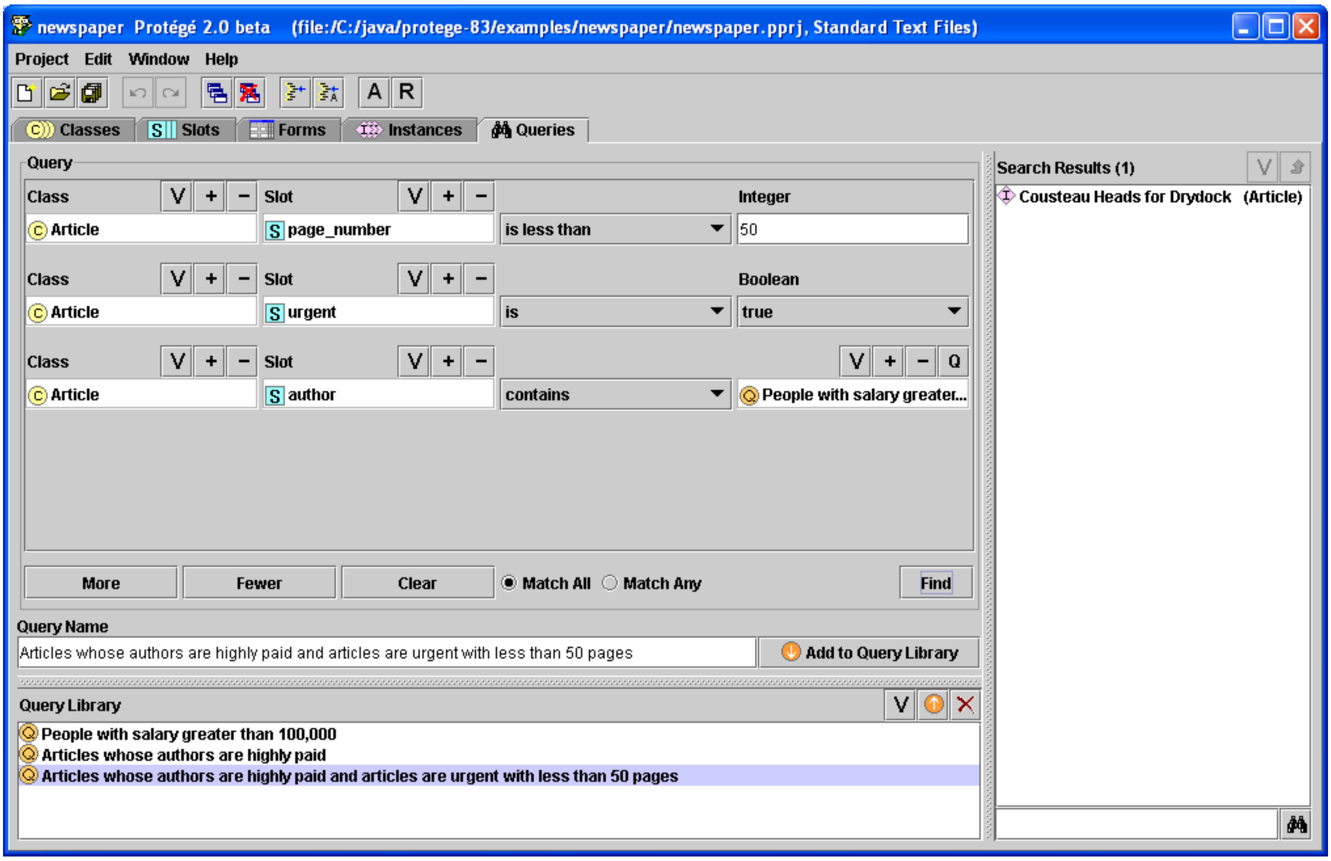


beckham Protégé 2.0 alpha (file:/D:/protege2.0a-54/owl-samples/beckham.pprj, JenaKnowledgeBaseFactory)

□ Grid □ Comment ☑ Inheritance ☑ Slot ☑ Inherited Slot ☑ Slot-Reference ☑ Restriction ☑ Restriction-Reference

Protégé / OWL / Plugins / Query Tab





newspaper Protégé 2.0 beta (file:/C:/java/protege-83/examples/newspaper/newspaper.pprj, Standard Text Files)

Query

Class	Slot		
Article	page_number	is less than	Integer: 50
Article	urgent	is	Boolean: true
Article	author	contains	People with salary greater...

● Match All ○ Match Any

Query Name: Articles whose authors are highly paid and articles are urgent with less than 50 pages

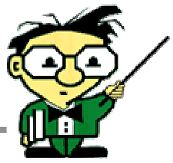
Query Library

- People with salary greater than 100,000
- Articles whose authors are highly paid
- Articles whose authors are highly paid and articles are urgent with less than 50 pages

Search Results (1)

- Cousteau Heads for Drydock (Article)

Protégé / OWL / Limitations and Future Work



- Major limitation: Only one namespace per project
  - Import currently not supported
  - Will become available with Protégé 2.1
- Very few OWL Language features are missing
  - Enumerated datatypes
- A-Box Reasoning: Given the following properties, to what classes does this individual belong
- Alternative syntaxes (e.g. “OWL for Dummies” formats)
- Database backend and Multi-User support are almost finished

Protégé / OWL / Summary

- Protégé has a long history of success in ontology development
- Protégé has an open, extensible platform for tools and applications
- The large user community ensures support and future evolution

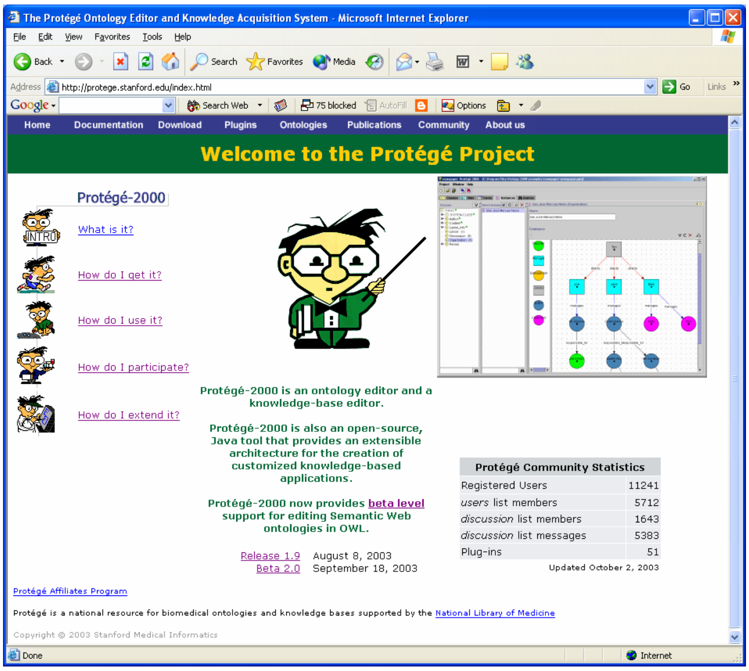
- The OWL Plugin is an extension of Protégé with editors, storage mechanisms, and access to reasoners
  - Integrated in the look-and-feel of Protégé
  - Providing optimized editing support for OWL specific language elements

- Third-party extensions exist (e.g. for visual editing)
- Many people are already using Protégé/OWL

→ **Get involved and contribute your ontologies and plugins!**

Protégé / OWL / More Information

- Protégé web site: http://protege.stanford.edu
- OWL Plugin: http://protege.stanford.edu/plugins/owl
- Getting help: protege-discussion list


The Protégé Ontology Editor and Knowledge Acquisition System - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://protege.stanford.edu/index.html
Go Links
Google Search Web 75 blocked AutoFill Options
Home Documentation Download Plugins Ontologies Publications Community About us
INTRO
Done Internet

Welcome to the Protégé Project

Protégé-2000

What is it?

How do I get it?

How do I use it?

How do I participate?

How do I extend it?

Protégé-2000 is an ontology editor and a knowledge-base editor.

Protégé-2000 is also an open-source, Java tool that provides an extensible architecture for the creation of customized knowledge-based applications.

Protégé-2000 now provides beta level support for editing Semantic Web ontologies in OWL.

Release 1.9 August 8, 2003
Beta 2.0 September 18, 2003

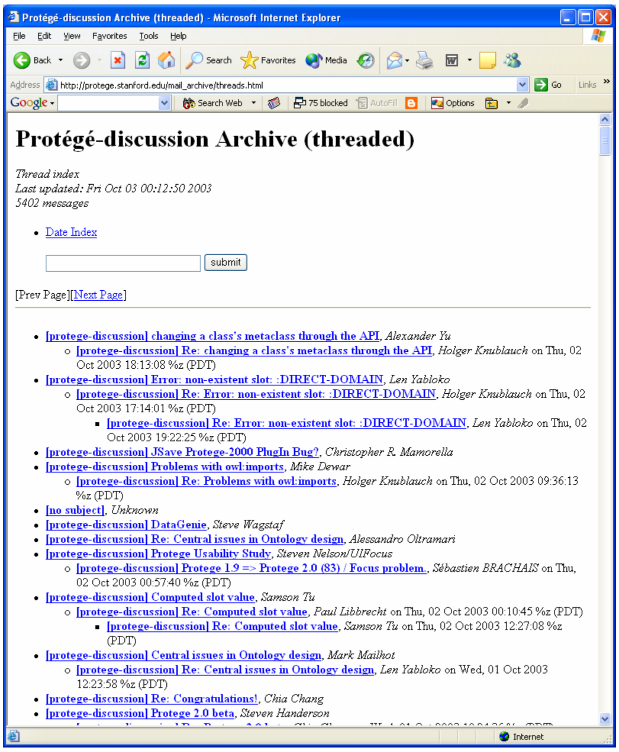
Protégé Community Statistics	
Registered Users	11241
users list members	5712
discussion list members	1643
discussion list messages	5383
Plug-ins	51

Updated October 2, 2003

Protégé Affiliates Program

Protégé is a national resource for biomedical ontologies and knowledge bases supported by the National Library of Medicine

Copyright © 2003 Stanford Medical Informatics


Protégé-discussion Archive (threaded) - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://protege.stanford.edu/mail_archive/threads.html
Go Links
Google Search Web 75 blocked AutoFill Options
Internet

Protégé-discussion Archive (threaded)

Thread index
Last updated: Fri Oct 03 00:12:50 2003
5402 messages

- Date Index

submit

[Prev Page][Next Page]

- [protege-discussion] changing a class's metaclass through the API, *Alexander Yu*
  - [protege-discussion] Re: changing a class's metaclass through the API, *Holger Knublauch* on Thu, 02 Oct 2003 18:13:08 %z (PDT)
- [protege-discussion] Error: non-existent slot: :DIRECT-DOMAIN, *Len Yabloko*
  - [protege-discussion] Re: Error: non-existent slot: :DIRECT-DOMAIN, *Holger Knublauch* on Thu, 02 Oct 2003 17:14:01 %z (PDT)
    - [protege-discussion] Re: Error: non-existent slot: :DIRECT-DOMAIN, *Len Yabloko* on Thu, 02 Oct 2003 19:22:25 %z (PDT)
- [protege-discussion] JSave Protege-2000 PlugIn Bug?, *Christopher R. Mamorella*
- [protege-discussion] Problems with owl:imports, *Mike Dewar*
  - [protege-discussion] Re: Problems with owl:imports, *Holger Knublauch* on Thu, 02 Oct 2003 09:36:13 %z (PDT)
- [no subject], *Unknown*
- [protege-discussion] DataGenie, *Steve Wagstaf*
- [protege-discussion] Re: Central issues in Ontology design, *Alessandro Oltramari*
- [protege-discussion] Protege Usability Study, *Steven Nelson/UIFocus*
  - [protege-discussion] Protege 1.9 => Protege 2.0 (83) / Focus problem., *Sébastien BRACHAIS* on Thu, 02 Oct 2003 00:57:40 %z (PDT)
- [protege-discussion] Computed slot value, *Samson Tu*
  - [protege-discussion] Re: Computed slot value, *Paul Libbrecht* on Thu, 02 Oct 2003 00:10:45 %z (PDT)
    - [protege-discussion] Re: Computed slot value, *Samson Tu* on Thu, 02 Oct 2003 12:27:08 %z (PDT)
- [protege-discussion] Central issues in Ontology design, *Mark Mailhot*
  - [protege-discussion] Re: Central issues in Ontology design, *Len Yabloko* on Wed, 01 Oct 2003 12:23:58 %z (PDT)
- [protege-discussion] Re: Congratulations!, *Chia Chang*
- [protege-discussion] Protege 2.0 beta, *Steven Handerson*