# Setting up & configurating Protege 4.x for COSMOS nmr-CV editing

## Scope of this Document

This is a short tutorial that aims at providing COSMOS participants and external interested parties with the necessary basic set-up information in order to open and manipulate the controlled vocabularies (CVs) developed in support of the COSMOS nmrML exchange standard. As COSMOS agreed to develop the CV in OWL syntax, this guideline focuses on the set-up of the open source ontology editor Protégé 4.x (P4). As Protégé is quite a complex tool, we recommend people just wanting to have a quick glance at the CV (without need to manipulate it) to open its HTML serialization in a normal web browser. The HTML version of the CV can be found under nmrML\docs\CVDocumentation .

The decision to represent the nmrML CV in OWL rather than OBO format was made for the following reasons:

* OWL is the W3C recommended standard for ontologies. It uis used widely in life sciences and beyond.
* A plethora of open source tools and libraries is available for OWL.
* OWL allows easy extension of formal semantics up to full description logics (DL).
* Major Top level ontologies (BFO, DOLCE) and bio-upper level ontologies (Biotop) are available in OWL format.
* OWL is based on XML and RDF, hence we foresee an easy migration towards future Open Linked Data approaches.
* W3C supported query languages like SPARQL and rule set-ups like SWRL exist that also integrate well with P4, e.g. are available as Protégé plugins.
* The Protégé editor is pretty robust compared to OBO Edit. Many free plugins are available covering most needs: <http://protegewiki.stanford.edu/wiki/Protege-OWL_4.0>

This document will be amended *as-we-go* according to the development stage of the CV and the skill level of the CV editors.

This document does **not** describe the CVs design principles. Documentations for these will be derived from <http://www.psidev.info/sites/default/files/CommunityPractice-revised.doc> where applicable.

This document does **not** describe the CVs term request pipeline. Documentations for these will be derived from the PSI 10 Minute guide to new CV terms currently found <https://github.com/nmrML/nmrML/blob/master/docs/CVDocumentation/PSI-Ten_minutes_guide_for_requesting_new_CV_terms.docx> We will only need such term request guideline once the first version has been released. We can look how the OBI ontology does it, but keep it simple and pragmatic.

## Contents

[Setting up & configurating Protege 4.x for COSMOS nmr-CV editing 1](#_Toc362357114)

[Scope of this Document 1](#_Toc362357115)

[Contents 1](#_Toc362357116)

[Download & Install Protégé 4.x 1](#_Toc362357117)

[Set-up Github reposiotory to get access to CV.owl file 1](#_Toc362357118)

[Open the CV in P4 1](#_Toc362357119)

[CV releases and versions 1](#_Toc362357120)

[Basic Orientation within GUI 1](#_Toc362357121)

[A few words on how P4 stores its set-up Preferences 1](#_Toc362357122)

[Set the class renderer to display human readable class labels in the hierarchy pane 1](#_Toc362357123)

[Set ID-scheme under File/preferences/New Entities 1](#_Toc362357124)

[Set CV namespace 1](#_Toc362357125)

[Recommended Class Naming Conventions 1](#_Toc362357126)

[Increase Java RAM Heap Size for P4 1](#_Toc362357127)

[Handling of Imports 1](#_Toc362357128)

[Term obsoletion practice and \_Purgatory usage 1](#_Toc362357129)

[Set Search to allow for full regular expression searches under File/preferences/general 1](#_Toc362357130)

## Download & Install Protégé 4.x

Download & Install the latest Java-based Protégé 4 form <http://protege.stanford.edu/download/download.html> , as described at <http://protegewiki.stanford.edu/wiki/Protege-OWL_4_FAQ#How_do_I_install_Protege-OWL.3F>

Start Protégé via its .exe or .bat file. A new default ontology will be created & displayed. Ignore it and open the latest nmrMLv.x.owl file in the Github at: nmrML\ontologies via File/Open.

Basic Instructions for using P4 is given here: <http://protegewiki.stanford.edu/wiki/Protege4GettingStarted>

## Set-up Github reposiotory to get access to CV.owl file

Install a github client, e.g. tortoise git for Windows. Check out the master repository from <https://github.com/nmrML/nmrML>

## Open the CV in P4

Before running the Protege 4, please check that there is a file catelog-v0001.xml in the ontology/branches directory. The file is needed for Protege 4 to find where the imported ontologies are. Start P4. Choose open owl ontology. Select and open the latest version of the nmrMLCV.owl from your local git directory, e.g. nmrML\ontologies\nmrMLv1.0.owl

## CV releases and versions

We should have something less complicated but related as the OBO community uses, e.g. <https://docs.google.com/document/d/14gVHOOZZaiUpCPUUXyG72WYaTN861-WsARGj2kuljjE/edit> . We discussed coupling git milestones to release versions. We also need to agree on unit tests to be done prior to a release.

## Basic Orientation within GUI

When an ontology is opened, P4 will open the first pane, the Active Ontology tab (to the left) per default, which will display the general metadata, import structure (<http://protegewiki.stanford.edu/wiki/Importing_Ontologies_in_P41>) and ontology metrics, pertaining to the whole representational artefact/CV rather than to singular representational units. For class editing you will use the Classes tab. Within a tab you can freely configure the GUI’s appearance by selected ‘views’ (<http://protegewiki.stanford.edu/wiki/Protege4Views>).

A general overview of how P4 can be configured to your general needs (Set preferences) can be found at <http://protegewiki.stanford.edu/wiki/Protege4Preferences> . All preferences specific to our nmrML CV set-up are explained in this document.

If you have any questions, please do not hesitate to contact me (DS) via email or the nmrML list at <https://groups.google.com/forum/?hl=en#!forum/nmrml/join>

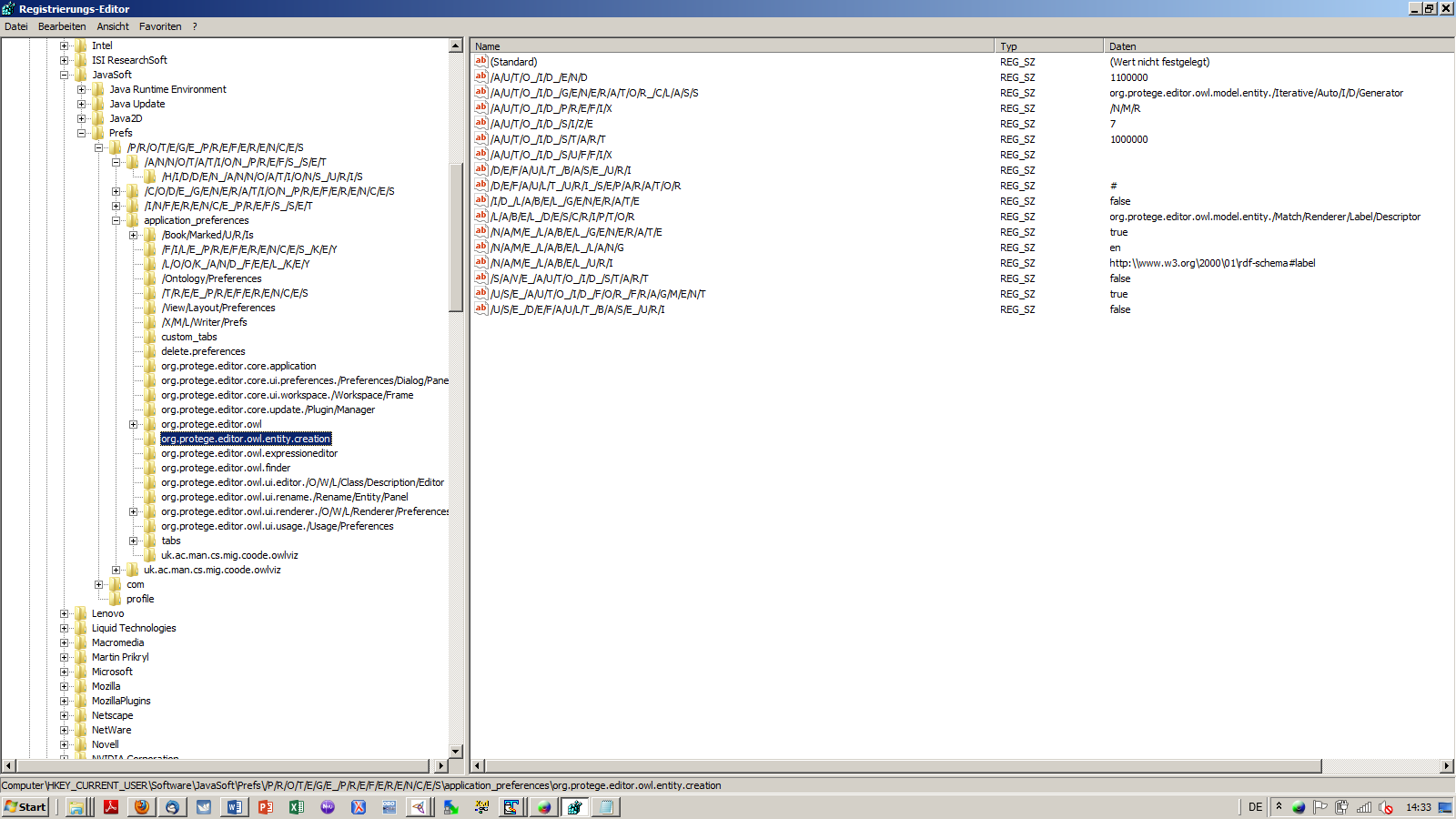
## A few words on how P4 stores its set-up Preferences

Protege 4 uses the Java Preferences API to store user preferences and activity. This allows for persistence of preference data across installations of Protege 4.x, e.g., recently opened ontologies, rendering preferences, location of Graphviz (used by the OWLViz plug-in), etc. The storage location of these preferences varies according to your operating system:

* Linux - stored in ~/.java/.userPrefs
* Mac OS X - stored in ~/Library/Preferences/com.apple.java.util.prefs.plist
* Windows - stored in the Windows Registry at HKEY\_CURRENT\_USER/Software/JavaSoft/Prefs

Occasionally, users may need to clear the Java Preferences to fix errors displayed by Protege. Preferences can either be deleted manually, or via a tool called the Java Preferences User Interface.

For example under Windows you can access and edit these preferences by typing “regedit” in the command line (Start/Run) and traverse the file path to the Prefs//P/R/O/T/E/G/E/\_/P/R/E/F/E/R/E/N/C/E/S folder. In the application\_preferences you will find a subfolder called HKEY\_CURRENT\_USER\Software\JavaSoft\Prefs\/P/R/O/T/E/G/E\_/P/R/E/F/E/R/E/N/C/E/S\application\_preferences\org.protege.editor.owl.entity.creation



In here, e.g. the ID setup for newly generated CV terms is stored. These preferences have been exported into a file named **protege\_entities\_prefs.reg** and have been placed in the github nmrML\ontologies folder.

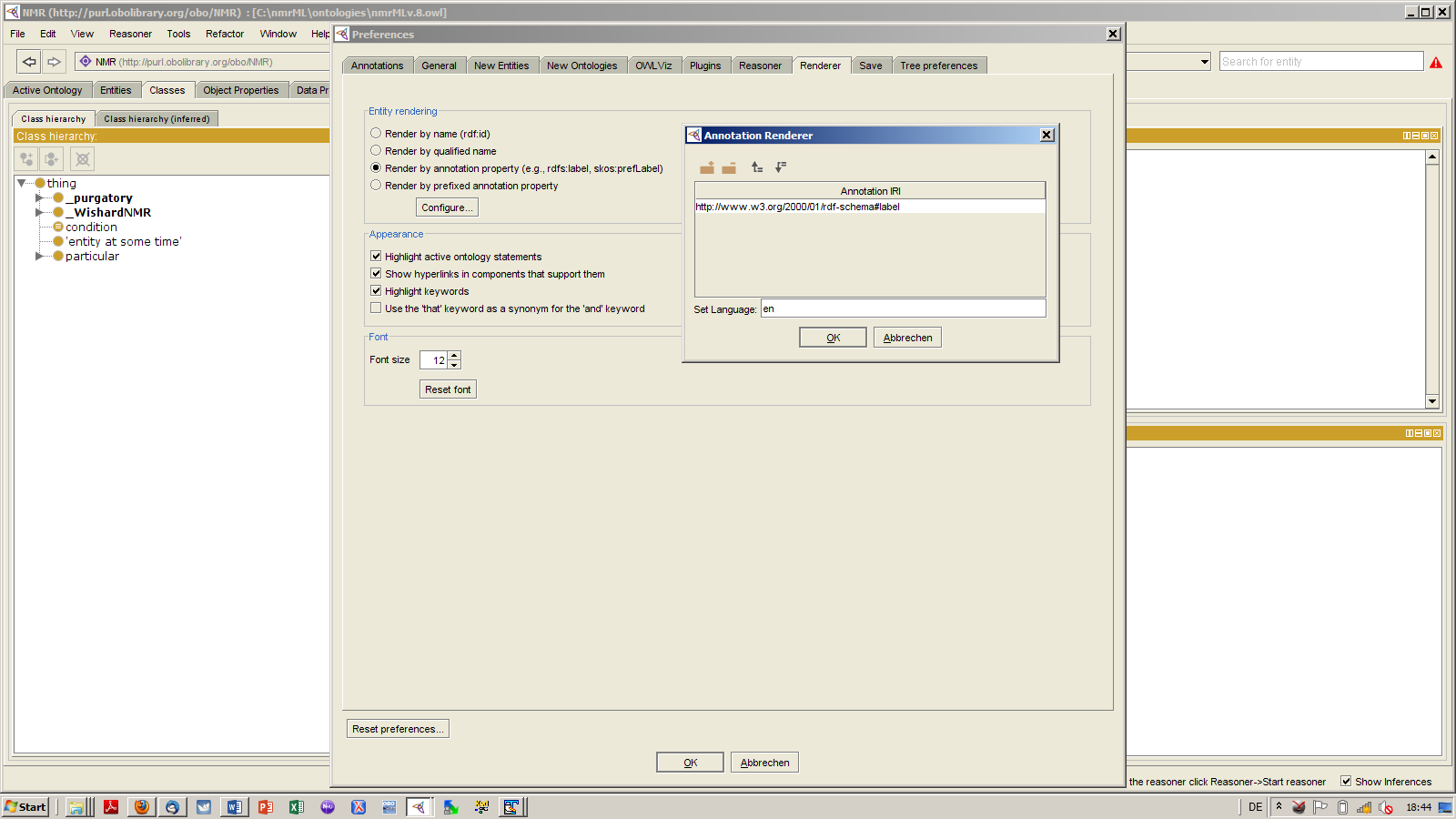
In windows you can now copy this protege\_entities\_prefs.reg file onto your desktop and execute it by double clicking on it. Your registry will then be automatically be updated with the new preference settings for the nmrML CV term ID generation in the appropriate registry location.

For all further preferences you proceed in the same way[[1]](#footnote-1). For more information look at <http://protegewiki.stanford.edu/wiki/ClearingP4Preferences>

As this is rather complicated, we recommend to just do the preference settings manually on your computer from within P4s preferences Tab (see below).

## Set the class renderer to display human readable class labels in the hierarchy pane

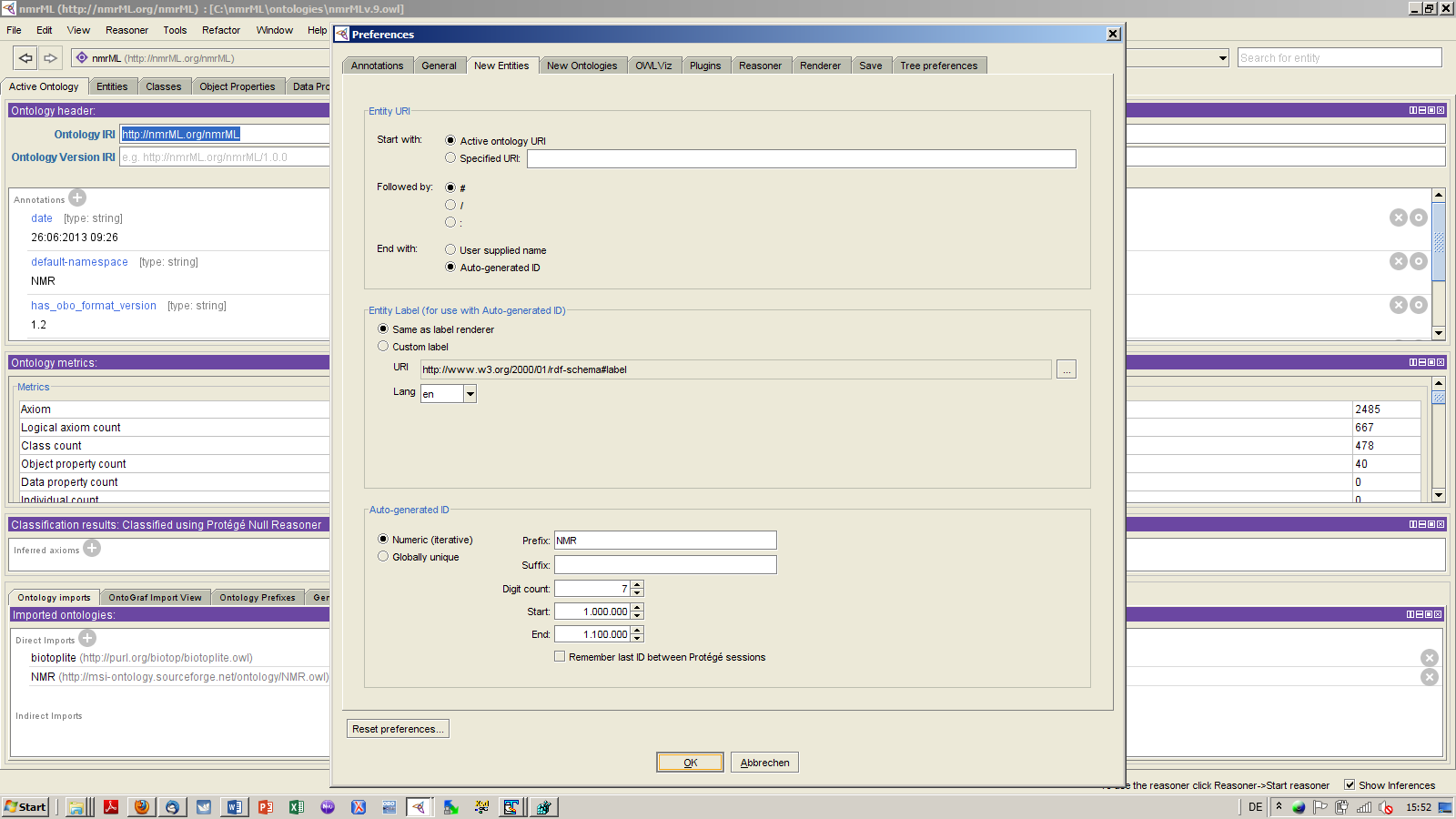
Set the renderer to use labels rather than the unintuitive IDs when displaying the class hierarchy (to the left of the GUI): Under File/preferences/Renderer tick the render by annotation property and hit “configure”. Then set it to display rdfs:label rather than rdf:about or rdf:id[[2]](#footnote-2). Set View/render by label. This will display the classes in the Class Tab/hierarchy pane by the classes rdfs:label, given one is provided. If no label is provided, the renderer will still display the class by its ID.



If classes are still displaying as IDs, go the the 'Renderer tab', click on the 'Annotations ...' button, select 'label', leave the language column blank and hit ok.

## Set ID-scheme under File/preferences/New Entities

Before editing the CV you need to make P4 aware of how you want a newly generated class/terms’ rdfs:about or rdf:id value to look like, i.e. you need to specify that we want semantics-free numeric IDs rather than human readable meaningful ones (to be captured by the rdfs:label field). For our nmrML CV, we should align the ID scheme to generate IDs like NMR:1000442 (Short NS prefix/IRI), colon, 7 digits, ranging from 1000000 to 1100000). This means we need to specify the CVs IRI (“NMR[[3]](#footnote-3)”), the following separator (“:”) and then we also need to specify the amount of digits the number should have and its numeric range. This is done in the File/Preferences/New Entities Tab:



This will generate the following registry entry in Windows Registry Editor Version 5.00

[HKEY\_CURRENT\_USER\Software\JavaSoft\Prefs\/P/R/O/T/E/G/E\_/P/R/E/F/E/R/E/N/C/E/S\application\_preferences\org.protege.editor.owl.entity.creation]

"/U/S/E\_/D/E/F/A/U/L/T\_/B/A/S/E\_/U/R/I"="false"

"/D/E/F/A/U/L/T\_/B/A/S/E\_/U/R/I"=""

"/D/E/F/A/U/L/T\_/U/R/I\_/S/E/P/A/R/A/T/O/R"="#"

"/U/S/E\_/A/U/T/O\_/I/D\_/F/O/R\_/F/R/A/G/M/E/N/T"="true"

"/N/A/M/E\_/L/A/B/E/L\_/G/E/N/E/R/A/T/E"="true"

"/I/D\_/L/A/B/E/L\_/G/E/N/E/R/A/T/E"="false"

"/L/A/B/E/L\_/D/E/S/C/R/I/P/T/O/R"="org.protege.editor.owl.model.entity./Match/Renderer/Label/Descriptor"

"/N/A/M/E\_/L/A/B/E/L\_/U/R/I"="http:\\\\www.w3.org\\2000\\01\\rdf-schema#label"

"/A/U/T/O\_/I/D\_/G/E/N/E/R/A/T/O/R\_/C/L/A/S/S"="org.protege.editor.owl.model.entity./Iterative/Auto/I/D/Generator"

"/A/U/T/O\_/I/D\_/S/T/A/R/T"="1000000"

"/A/U/T/O\_/I/D\_/E/N/D"="1100000"

"/A/U/T/O\_/I/D\_/S/I/Z/E"="7"

"/A/U/T/O\_/I/D\_/P/R/E/F/I/X"="/N/M/R"

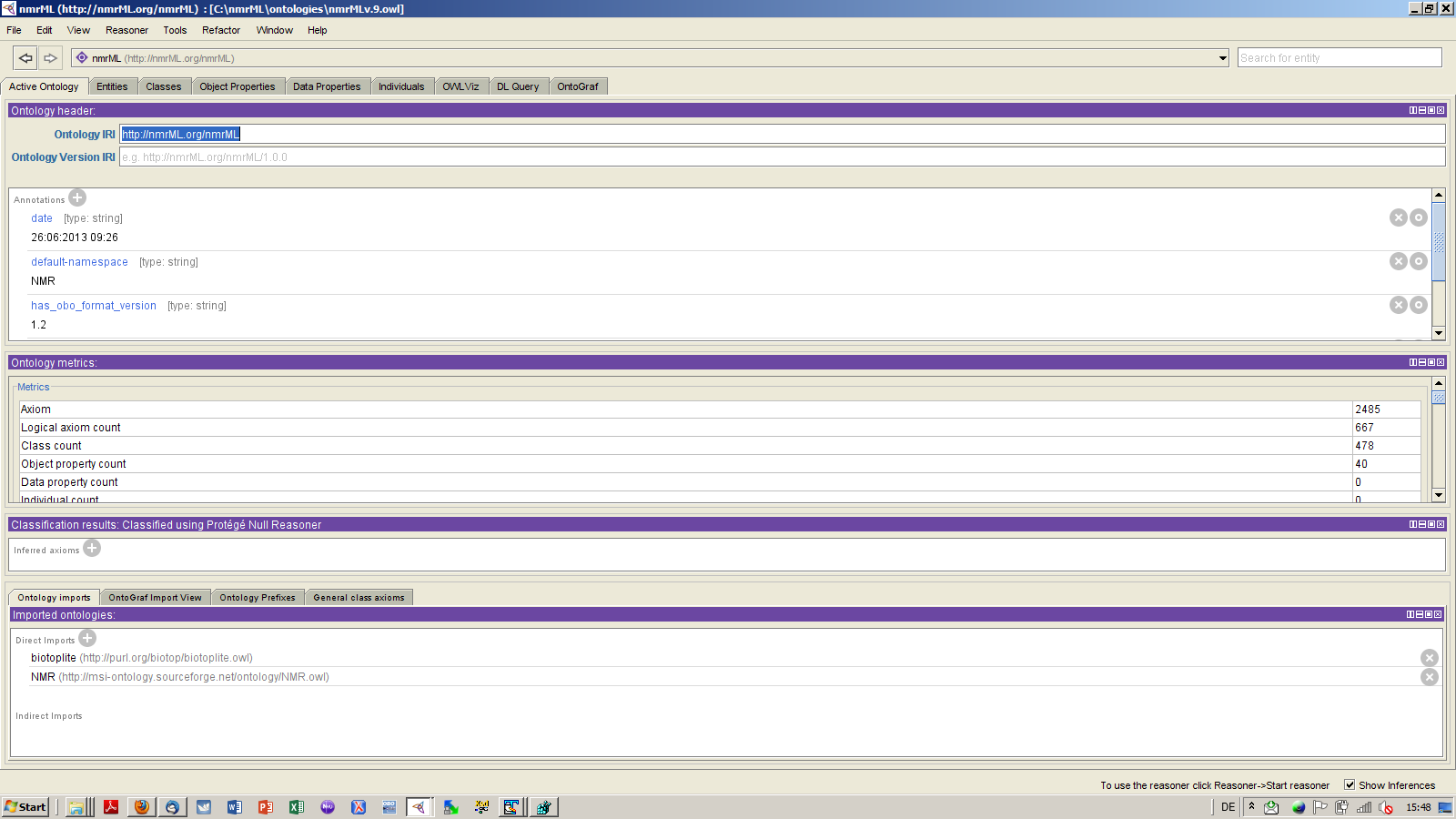
"/A/U/T/O\_/I/D\_/S/U/F/F/I/X"=""

"/S/A/V/E\_/A/U/T/O\_/I/D\_/S/T/A/R/T"="false"

"/N/A/M/E\_/L/A/B/E/L\_/L/A/N/G"="en"

## Set CV namespace

In the Active Ontology pane/Ontology header/Ontology IRI field we put <http://nmrML.org/nmr>ML [[4]](#footnote-4) to set the active ontology’s namespace in accordance to the one of the XSD (xmlns:nmrML=http://nmrML.org/nmrML ).



## Recommended Class Naming Conventions

To provide some guidance on how a meaningful and unambiguous term /class label should look like, the OBO Foundry has provided a set of naming conventions (<http://obofoundry.org/wiki/index.php/Naming> ). Within P4 we can define labelling Checks via the P4 plugin OntoCheck: <http://www.imbi.uni-freiburg.de/ontology/OntoCheck>

## Increase Java RAM Heap Size for P4

To open larger and complex ontologies you should increase the RAM heap size to be >3GB. This is explained at <http://protegewiki.stanford.edu/wiki/Setting_Heap_Size>

## Handling of Imports

There is a configuration file, catalog-v001.xml, which sets Protege to load imported ontologies from local files.

## Term obsoletion practice and \_Purgatory usage

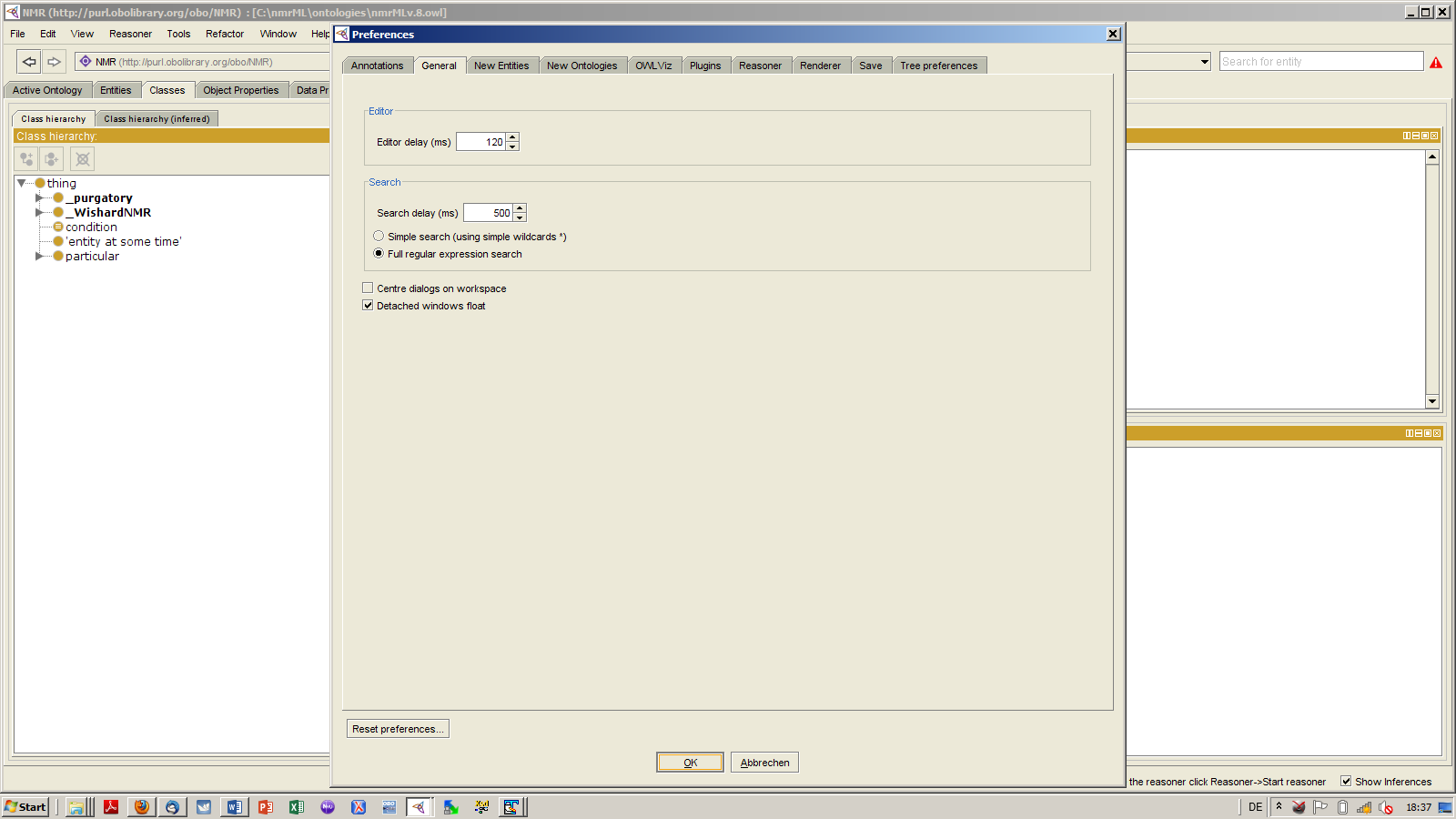
There are multiple ways to do this. The easiest would be using owl: Deprecated Class, owl: Deprecated Property. Alternatively we can re-use the policy specified for OBI at <http://obi-ontology.org/page/OBIDeprecationPolicy>.

Deprecated terms should be advertised at the nmrML mailing list prior to deprecation. The terms can be deprecated if no objections are raised within a week.

Step by step (taken for OBI, so ignore annotation property usage for the moment):

1. remove any child of term-to-be-deprecated to sensible parent, drag and drop one by one
2. remove all logical axioms of term-to-be-deprecated, you can copy axioms to the editor notes as record
3. make sure no any other terms use term-to-be-deprecated, check usage tab
4. add reason for deprecation property to the term-to-be-deprecated by adding annotation 'has obsolescence reason', choose predefined terms (e.g. 'failed exploratory term' , 'placeholder removed' , 'terms merged' , 'term imported' , 'term split') from 'Entity IRI' tab's 'Individuals' tab and add replacement term property if there is one
5. add prefix to rdfs label obsolete\_
6. remove from existing parent class
7. add annotation 'deprecated' with value "true" and type "boolean"
8. save
9. reason over in P4, check consistency - if you leave the old parent and the obsolete one will generate an inconsistency
10. if OK check in as a new file with an git log

## Set Search to allow for full regular expression searches under File/preferences/general



1. I am hesitant to recommend to export the whole HKEY\_CURRENT\_USER\Software\JavaSoft\Prefs\/P/R/O/T/E/G/E\_/P/R/E/F/E/R/E/N/C/E/S folder, as it may contain country and language specific setting not applicable to your local environment. [↑](#footnote-ref-1)
2. On the difference, look at <http://answers.semanticweb.com/questions/2189/should-i-use-rdfabout-or-rdfid> [↑](#footnote-ref-2)
3. Should we use the full URL path here ? E.g. <http://nmrML.org/nmrCV>, or <http://nmrML.org/nmrML> ? Read more on <http://protegewiki.stanford.edu/wiki/Protege4NamingAndRendering> [↑](#footnote-ref-3)
4. Or should we rather than having the same NS for both CV and XSD, have an own one for the CV, e.g. <http://nmrML.org/nmrCV> ? [↑](#footnote-ref-4)