# Lab exercise Protégé

**Aim**
After completing this lab you should be familiar with the main functionality of a modern ontology development tool. You will learn how to download, update, and maintain a simple ontology. Furthermore, you will learn how to interact with the ontology in a programming environment by using the Java-based Jena library. This is a rather simple lab.

## 1. Ontology development

The first task in the lab is to edit an ontology in Protégé.

Protégé is a free, open source ontology editor. It provides a suite of tools to construct domain models and knowledge-based applications with ontologies. Protégé ontologies can be exported into a variety of formats. In the lab we use OWL.

OWL Web Ontology Language is intended to provide a language that can be used to describe the classes and relations between them that are inherent in Web documents and applications. OWL Web Ontology Language Guide, a recommendation from W3C, demonstrates the use of the OWL language. More information about OWL from W3C can be found here.

If you want to setup Protégé on your personal computer you should download version 3.4.1 (build 537) and select ”Basic + OWL” during the installation process (in the Choose Components step). Also note that you can download versions without a bundled VM if you already have one installed (most likely you do). There is a version 4 of Protégé, but it has been changed in major architectural ways and the ontology you will work with in this lab is not compatible with it. Apart from these tips we cannot assist in troubleshooting any problems you might encounter setting up Protégé on your personal computer.

### Edit an ontology in Protégé 3.4.1

On the Protégé user documentation page you can find links to tutorials and presentations that should help you get started with Protégé, but you should be able to complete the exercise without consulting them thanks to Protégé's intuitive user interface.
Download a university ontology uni.zip. Edit the ontology in Protégé.

- Open the ontology by opening the existing project *university.pprj*. Note that Protégé doesn't understand network paths, so instead of using *neptunus/<yourusername>*, you should use its alias *Z:* when specifying the path to the file.
- Locate the class *Student* and create a sub-class *UndergraduateStudent* for it.
- Create a new class *GraduateStudent*, it should also be a sub-class of *Student*.
- Create a new ObjectProperty *takesCourse*. Set its domain to *Student* and its range to *Course*.
- Create a new DatatypeProperty *age*. Set its domain to *Person* and its data type to *int*.
- Create an instance for *UndergraduateStudent*, e.g. *Harry Potter*. Give his/her age, emailAddress and courses he/she takes. For the courses he/she takes, you may have to create the courses first, i.e., instances of *Course*.
- Extend the ontology as you wish.
- Output an OWL file for your ontology.

## 2. Ontology use

Jena provides a popular programmatic environment for RDF, RDFS, and OWL. Protégé also uses Jena for various tasks. In the second part of the lab you will write a simple Java program that demonstrates how to work with an OWL ontology using Jena. The program should print out a tree of all classes along with any instances a given class might have. We have prepared a lab skeleton for you in the form of an Eclipse project so you don't have to write the program from scratch. The Eclipse project can downloaded here. It should be unpacked and imported into Eclipse. The project contains two Java source files, ClassHierarchy.java and Main.java, and you need to study them and complete the missing parts for the program to work as described above. There are TODO-markers visible in the margin to help you find the parts where your code should go. The program should operate on your university ontology so you will need to specify the path to your .owl-file as well.

Javadoc should be available for the Jena library in the Eclipse project but if you want more documentation you could consult the Jena Documentation page.

A few notes about Eclipse. There are a few different Eclipse versions installed on the lab computers, make sure you select *Java->eclipse-java-galileo-SR1-win32* under All Programs in the start menu. When you launch Eclipse for the first time it will ask where your workspace should be. Accept the suggested one (which is located on your network-mounted drive) and tell Eclipse not to ask again. After selecting a workspace you should find yourself on the Welcome Screen. If you close the Welcome Screen, you will be taken to the Java perspective and that's where you want to be for the next step when it's time to import the lab skeleton as an Eclipse project. First unpack it where you like, in your workspace folder for example. Then, make sure you are in the Java perspective and either right-click the Package Explorer (left-hand side) and select “Import...” or select “Import...” from the file menu. In the dialog that appears you want to select “General->Existing Projects into Workspace”, press “Next >” and browse to the directory where you placed the project and Eclipse should be able to find it. The name of the project is “Theme 5 Jena”.

If you don't want to use Eclipse for this task but some other IDE or simply a text editor and invoke the Java compiler from the command line you are free to do that, but we cannot offer any support. However, one thing you need to make sure if you want to build the program without using Eclipse is that the Jena .jar-files are available to the builder, or the program won't build due to unresolvable imports. If you use Eclipse, that has already been setup for you in the project. You should download version 2.6.2 of Jena, which is the latest version at the time of writing.

To demonstrate the lab, please hand in a lab report containing a complete print-out of the code in your Eclipse project and also include a print-out of what it displays when it's run.