

# OWLviz – A visualisation plugin for the Protégé OWL Plugin

## 1 About OWLviz

OWLviz is designed to be used with the Protege OWL plugin. It enables the class hierarchies in an OWL Ontology to be viewed and incrementally navigated, allowing comparison of the *asserted* class hierarchy and the *inferred* class hierarchy. OWLviz integrates with the Protege-OWL plugin, using the same colour scheme so that primitive and defined classes can be distinguished, computed changes to the class hierarchy may be clearly seen, and inconsistent concepts are highlighted in red. OWLviz has the facility to save both the asserted and inferred views of the class hierarchy to various concrete graphics formats including png, jpeg and svg.

OWLviz was developed as part of the CO-ODE project ([www.co-ode.org](http://www.co-ode.org)) by Matthew Horridge at the University Of Manchester. It is licensed under the Lesser GNU Public License. Copyright The University Of Manchester 2004. OWLviz uses the layout algorithms provided by the GraphViz software by AT&T, and also uses the Batik libraries by the Apache Software Foundation.

## 2 Installing OWLviz

You *must* have pre-installed:

- **The very latest version of Protégé 2.1 (currently in beta stage)**
- The **latest** version of the **Protégé-OWL plugin**.
- **GraphViz** version 1.10 or later  
(available from <http://www.research.att.com/sw/tools/graphviz/download.html>)  
(Although it is not necessary, is recommended that GraphViz is installed in the default location suggested by the GraphViz installer).

To install OWLviz, simply unzip the OWLviz.zip file into the **Protégé Plugins** folder, which will create folder named 'uk.ac.man.cs.mig.coode.owlviz' containing the necessary jar files.

If GraphViz is not installed in the default location, then you will need to specify the location of the GraphViz DOT application – instructions for this can be found in section 4.

## 3 Using OWLViz

### 3.1 The OWLViz User Interface

The OWLViz user interface consists of the following parts:

#### 3.1.1 The OWLViz toolbar

The OWLViz toolbar is shown in Figure 1. All the functionality offered by OWLViz may be accessed via this toolbar.

**Figure 1 - The OWLViz Toolbar**



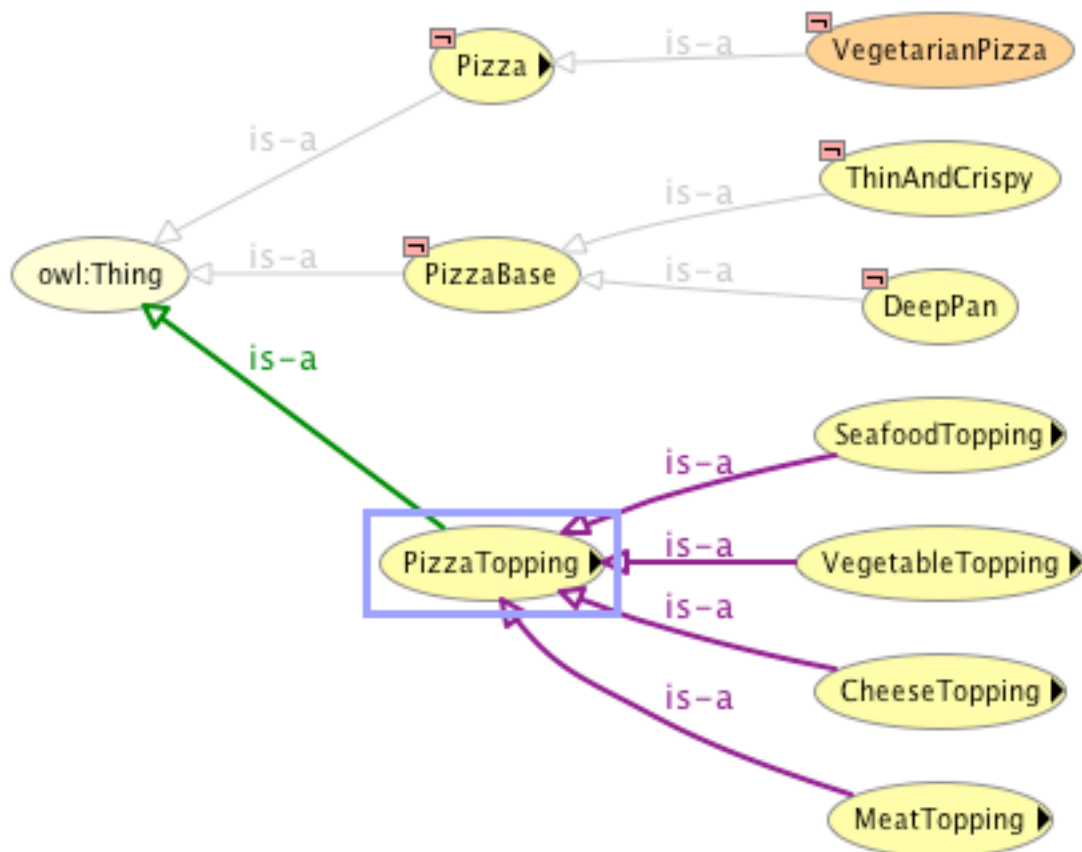
From left to right, the buttons are as follows:

- Show class
- Show subclasses
- Show superclasses
- Hide class
- Hide subclasses
- Hide superclasses
- Hide classes past a specified radius
- Hide all classes
- Display only primitive classes
- Zoom out
- Zoom in
- Export to graphics format
- Options

#### 3.1.2 The Asserted Model and Inferred Model Tabs

OWLViz makes it possible to view all or part of the asserted and inferred class hierarchies. The asserted and inferred class hierarchies appear on different tabs (Asserted Model for the asserted class hierarchy and Inferred Model for the inferred hierarchy). Unless the ontology that is being edited has been classified at least once, the classes in the inferred hierarchy will be displayed without any edges connecting them – this is because the edges represent inferred subsumption relationships, which do not exist until classification has taken place. Each tab contains a scrollable ‘main’ view, in which the classes being displayed appear as directed graphs as shown in Figure 2, which shows part of the class hierarchy from a Pizza ontology.

Figure 2 - The Main Class Hierarchy View



### 3.1.3 Class Hierarchy Tree Views

On both the asserted and inferred model tabs there are tree views of the class hierarchy (like the traditional tree view on the Protégé-OWL classes tab). These tree views make it possible to select any class in the current ontology. If the selected class is being displayed in the OWLViz hierarchy views, then the selection is indicated by a selection box that is drawn around the class. Many of the functions such as 'Show Class', 'Show Subclasses' etc. relate to the selected class.

### 3.1.4 Thumbnail Views

Both the inferred model and asserted model tabs contain thumbnail views, which are useful if the displayed class hierarchies are too big to fit on the screen. The red box on each thumbnail view reflects the portion of the main class hierarchy views being displayed, and can be dragged around to reposition the main views.

## 3.2 The Class View Colouring and Icons

### 3.2.1 Colouring Of Classes

As can be seen from Figure 2, the colour coding of classes is the same as that used by the Protégé-OWL plugin. Primitive classes (classes that have no equivalent classes – or definitions) are coloured yellow (system classes and non-editable classes such as owl:Thing are shown in a paler yellow). Defined classes (classes that have at least one equivalent class) are coloured in orange. If the option to show hidden and system classes has been selected for the Protégé Project, then the graph is also able to show system classes and meta classes, which are displayed in a green colour as they are in Protégé.

### 3.2.2 Selected Classes

The main graph view supports class selection. In Figure 2, the 'PizzaTopping' class is selected, which is indicated by the surrounding blue box. Edges between classes are coloured light grey, unless they join the selected class with another class, in which case, super class edges are shown in green, and subclass edges are shown in purple.

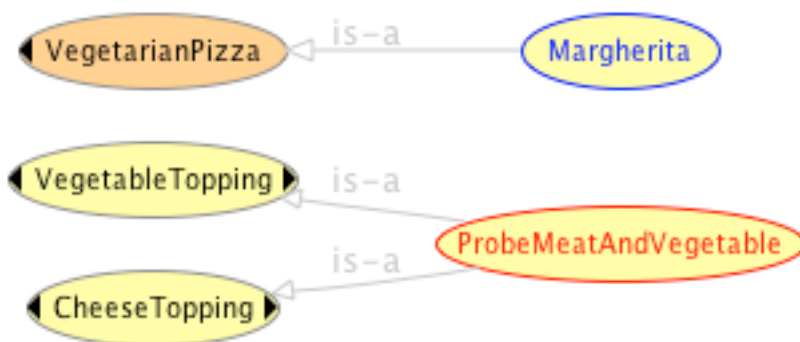
### 3.2.3 Disjoint Classes

A small 'disjoint class' icon is drawn next to a class if it is disjoint with the selected class – for example, in Figure 2, the classes Pizza, VegetarianPizza, PizzaBase, ThinAndCrispy and DeepPan are disjoint to the selected class PizzaTopping. Note that 'inherited disjointedness' is shown, so although ThinAndCrispy and DeepPan are not explicitly stated to be disjoint to PizzaTopping, they are in fact disjoint because their super class PizzaBase is disjoint the PizzaTopping.

### 3.2.4 Reclassified And Inconsistent Classes

After running a classifier, classes may have been reclassified or have been found to be inconsistent. If a class has been reclassified, then it's border and label will be coloured blue as shown in Figure 3, where Margherita (a kind of Pizza with only Mozzarella and Tomato toppings) has been reclassified as a subclass of vegetarian pizza. If a class has been found to be inconsistent, then it's border and label are coloured red as shown in Figure 3, where 'ProbeMeatAndVegetable' has been found to be inconsistent (because it's a subclass of Vegetable and Cheese Toppings which are disjoint to each other).

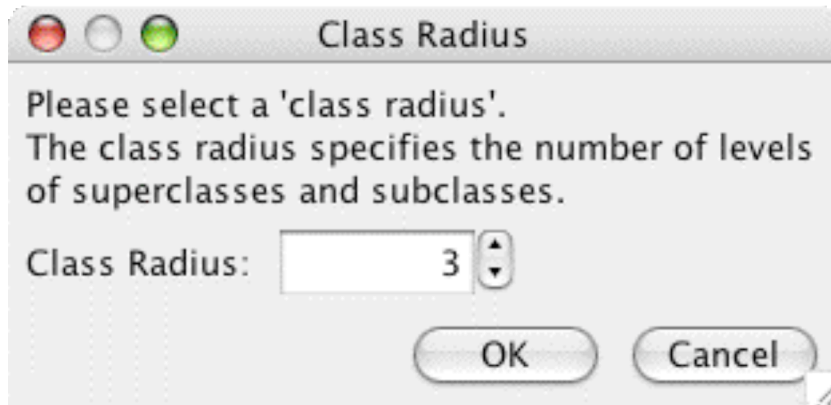
Figure 3 - Class Colouring



### 3.3 Displaying a Class

To display a class, first select the class to be displayed in the tree view. Next click the Show class button on the OWLViz toolbar. This will pop up a 'class radius' dialog box as shown in Figure 4, which is used to specify a radius that indicates how many levels of superclasses and subclasses should be shown around the selected class. For example, a class radius of 1 would indicate that the selected class and it's super classes and sub classes should be displayed. If the class radius is set to zero, then only the selected class is displayed

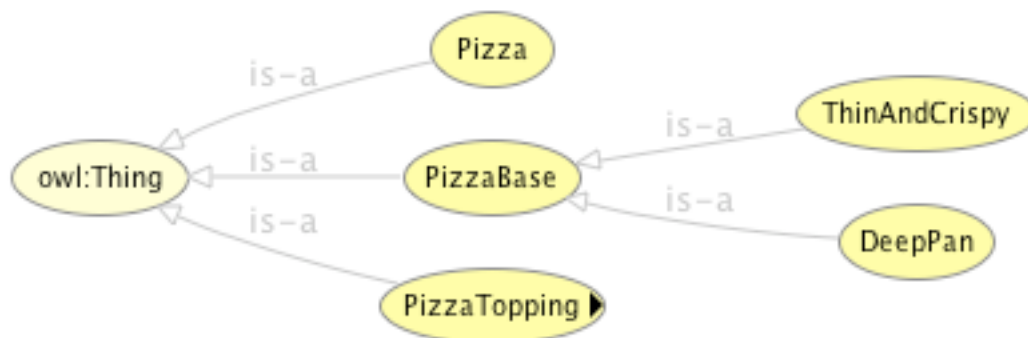
**Figure 4 Class Radius Dialog**



### 3.4 Incrementally Navigating the Class Hierarchy

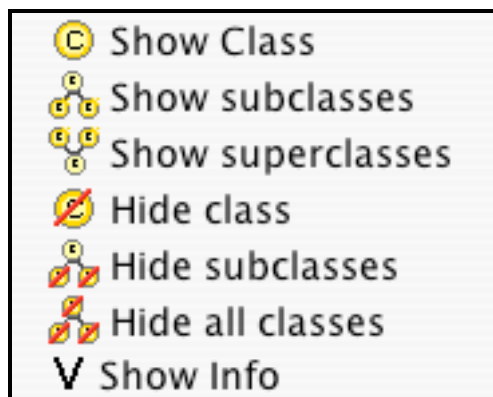
The class hierarchy may be incrementally navigated by expanding and collapsing super classes and sub classes of any class. If a selected class does not have all of its super classes or sub classes displayed, then black 'expansion arrows' are drawn to indicate this. For example, Figure 5 shows part of the class hierarchy from a pizza ontology. The small black expansion arrow on the 'PizzaTopping' node indicates that PizzaTopping has some sub classes that are not shown.

**Figure 5 Expansion Arrows**



The subclasses of PizzaTopping may be shown by selecting PizzaTopping by clicking on it, then either pressing the 'Show subclasses' button on the toolbar, or by right clicking (ctrl + click on the Mac) and then selecting the 'Show subclasses' item from the menu (shown in Figure 6). Other commands such as 'Show superclasses', 'Hide subclasses' etc. may be used in a similar way. The Show Info command seen on the context menu in Figure 4 displays the traditional Class Information dialog box in Protégé.

**Figure 6 - Context Menu**



A particularly useful feature is the 'Hide classes past a specified radius' command, which can be accessed via the toolbar button with the following icon.



This hides super classes and sub classes of the selected class that are beyond a radius specified (using a popup dialog similar to that shown in Figure 4). This command is useful if quite a lot of classes have been added to the graph, and the graph needs to be refocused around the selected class.

### **3.5 Displaying Information About the Selected Class**

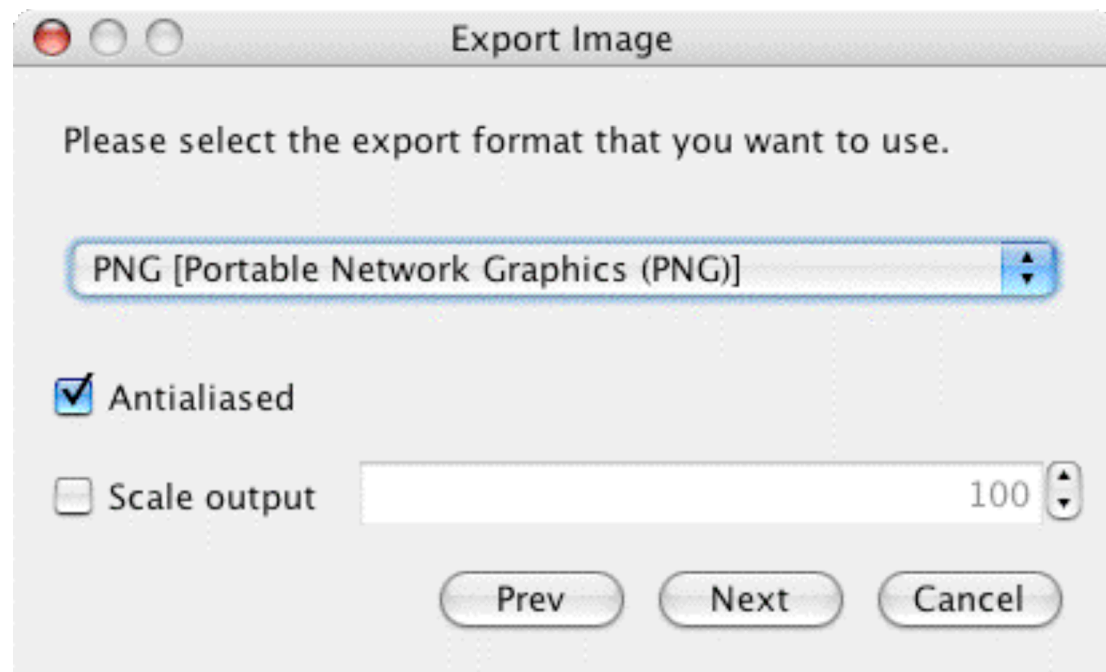
When a class is selected in the main graph view, it can be double clicked with the left mouse button to display a Protégé popup window that contains the information found on the OWLClasses tab.

### **3.6 Exporting The Class Hierarchies To Images**

Both the asserted and inferred class hierarchies may be exported to concrete image formats using the export image wizard, shown in Figure 7. This may be accessed via the toolbar using the 'export image' button that has the following icon.



Figure 7- Export Image Wizard

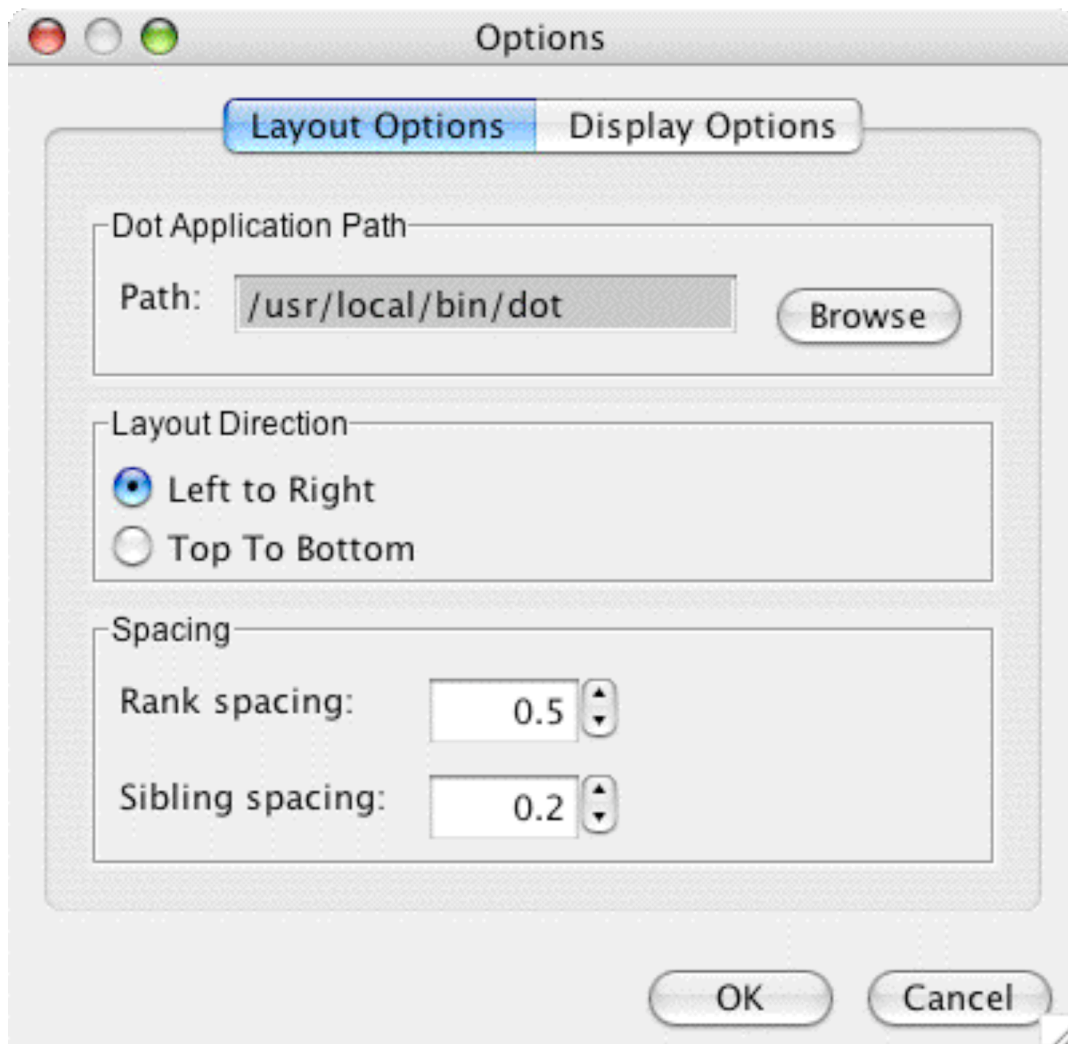


## 4 Setting The Path To DOT

OWLViz uses the GraphViz layout algorithms contained in the DOT application. If you do not have the GraphViz DOT application installed in the default location (/usr/local/bin on Mac OS X, Linux, and Unix or C:\Program Files\ATT\GraphViz\bin) then you will need to tell OWLViz where to find it. This can be done using the options dialog shown in Figure 8, which can be accessed using the options button on the OWLViz toolbar. If you are using OWLViz on a Mac and have the PixelGlow (<http://www.pixelglow.com/graphviz/>) version of GraphViz installed you can point to the PixelGlow GraphViz application and OWLViz will find the dot application that is embedded in this.



Figure 8 - OWLViz Options Dialog



## 5 Acknowledgements

Thanks to **Holger Knublauch** for developing the Protégé OWL plugin!

The AT&T team who developed GraphViz

This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>). Please see the Batik license enclosed with the distribution of OWLViz.