

Requirements: Building a web tool for OWL visualization

The focus of the project is to allow users to create a knowledge graph in a simple way by:

- (1) using visual language elements and
- (2) RDF search engine

The tool will cover all OWL lite axioms by using and extending the VOWL notation as in the WebVOWL tool and finding and suggesting entities to be created on other knowledge graphs using RDF search engine. In such a way, KG will be easily created and linked to one or more KG databases or RDF stores.

Development details:

This tool is similar to WebVOWL and RDFExplorer which are open source tool available online, So you can reuse 80% of the code required from both the tools.

Tool links: <http://www.visualdataweb.de/webvowl/>, <https://www.rdfexplorer.org/>

Git links: <https://github.com/hvarg/RDFExplorer>, <https://github.com/VisualDataWeb/WebVOWL>

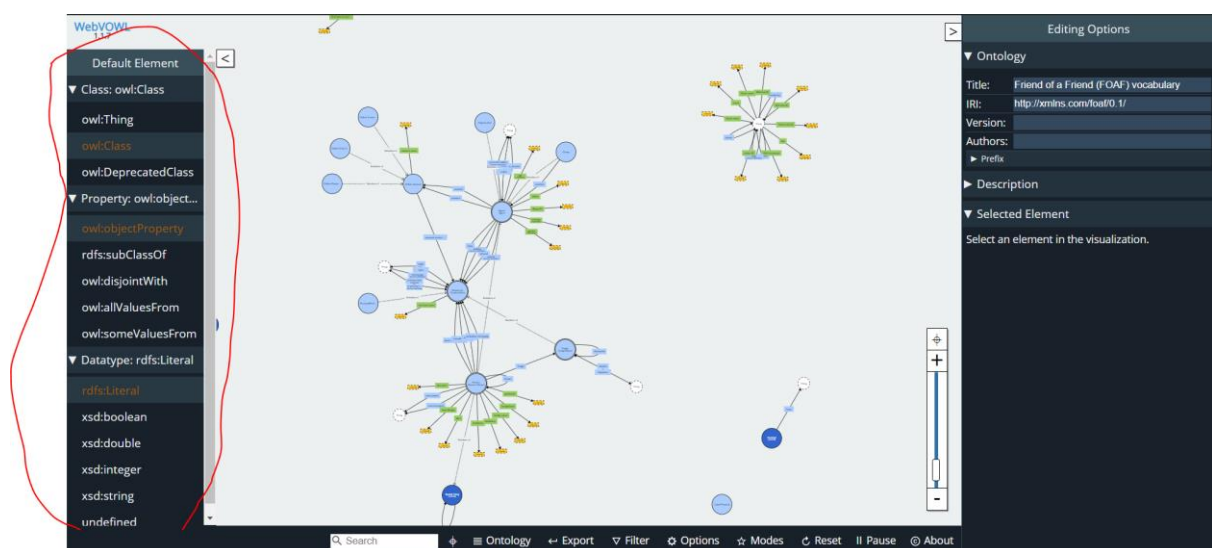
The tool will be divided into 4 sections: on top, left, middle, and right.

Top section:

On-top-side, the user will be asked in which domain he will create the KG in order to select the appropriate knowledge graphs to use e.g. bioportal or wikidata.

This will be a dropdown that enables the user to select from bioportal or wikidata.

Left side bar:



The left-side will have the main icons and terms for KG development. The included icons show VOWL notations, e.g., circle for “owl: Class”, an arrow for “owl:subClassof”, etc.

As shown in the image, the WebVOWL uses the names of the elements, in this tool that has to be replaced with shapes denoting each element.

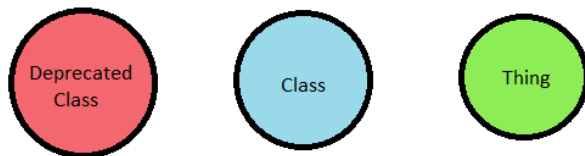
1. Class section:

Owl:Class	-	Blue circle as shown in WebVOWL tool.
Owl:Thing	-	Green circle
Owl:DeprecatedClass	-	Red Circle as in WebVOWL

These symbols to be added under the **Class** toggle in the left side bar.

User can drag these elements from the left side bar to the svg section in the middle.

This drag and drop working is similar to the one in RDFExplorer. Code reuse is possible.



2. Property section














These are similar to the properties in WebVOWL but more additions are done.

The symbols stay the same as below. These symbols to be added under **Property** toggle in the leftside bar.

User can click on a property and connect 2 classes, working similar to that of webVOWL using the arrow. The selected property should be added along with the symbol shown below between the 2 classes.

AreSame (as)	$(c, as, c) \mid (i, as, i) \mid (hr, as, hr) \mid (f, as, f) \mid (t, as, t) \mid (s, as, s)$		owl:equivalentClass, owl:equivalentProperty
AreDifferent (ad)	(i, ad, i)		-
AllDifferent (ald)	$(i, \{i\}, ad, i, \{i\})$		-
IsA (isa)	$((i \mid c), isa, c) \mid (hr, isa, hr) \mid (f, isa, f) \mid (t, isa, t) \mid (s, isa, s)$		rdfs:subClassOf, rdfs:subPropertyOf
InverseOf (io)	$(hr, io, hr) \mid (f, io, f) \mid (t, io, t)$		owl:inverseOf
HasValue (hv)	$(c, hv, \langle String \rangle) \mid (i, hv, \langle String \rangle)$		owl:DatatypeProperty
HasRelation (hr)	$(c, hr, c) \mid (i, hr, i)$		owl:ObjectProperty
Functional (f)	$(c, f, c) \mid (i, f, i)$		owl:inverseFunctional-Property, owl:Functional-Property
Transitive (t)	$(c, t, c \text{ (*The same one*)}) \mid (i, t, i)$		owl:TransitiveProperty
Symmetric (s)	$(c, s, c) \mid (i, s, i)$		owl:SymmetricProperty

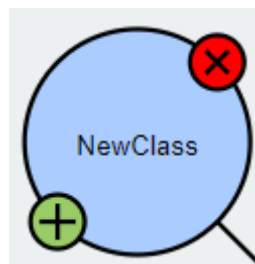
This is just an example of what we had created earlier, but then decided on to including them similar to the display of WebVOWL tool.

 Class	 Are same	 Is a	 Has value	 Transitive
 Individual	 Are different	 Inverse of	 Has relation	 Symmetric
	 All different	 Intersection	 Functional	

3. Datatype section:

Same as WebVOWL tool. No need of separate shapes. You can follow the exact notations and working like the WebVOWL tool.

- User can select a data type at first
- Click on the green '+' button
- Creates a new data type property with the selected data type.

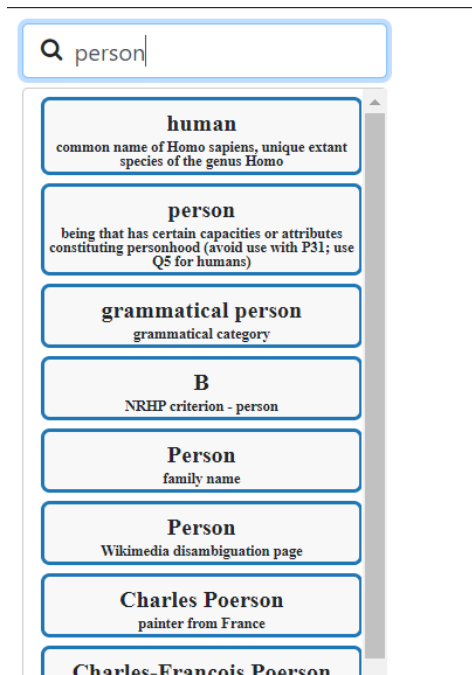


Middle section:

The Middle section will show the actual visualization area of the creation of the knowledge graphs.

Main features to be added here:

1. Person drags a new class to this area. On clicking the text, in case of WebVOWL, the user is allowed to change the text to a new name. In this tool we require including the functionality of search similar to RDFExplorer.
2. When a user starts typing inside, the search has to be triggered to the data store that is selected in the drop down (Wikidata or bioportal) in the top section. When a user selects a particular name from the option, this has to be assigned to the new class and the properties belonging to that entity has to be displayed in the rightside bar. Will be explained in the next section. RDFExplorer uses Wikidata and its APIs.
3. An image of the search operation in RDFExplorer is given below, they have included it in the left side bar. The code is available in their github which can be reused. The same has to be done using bioportal also. If a user selects bioportal those item names from bioportal has to be displayed.



Right side:

As explained above, similar to the RDFExplorer, when a user clicks a particular entity, its datatype and object properties obtained from the API has to be displayed similar to RDFExplorer. A user can also select from these existing properties and use the arrow among classes to add these properties. An orange box instead of the violet box can be used for these relations.

The idea is when user starts creating a class and start typing, the tool searches at the same time of the concepts in KG using a search engine and shows all information related on the right side where he can reuse them.

Person

Person

Datatype properties [5]

Commons category : Person (surname)

native label :

Soundex : P625

Caverphone : PSN111

Cologne phonetics : 1786

Object properties [4]

instance of
 Q ▼

said to be the same as
 Q ▼

writing system
 Q ▼

different from
 Q ▼

This created graph has to be converted to OWL ontology format. The code is available in the export -> export as TTL section in WebVOWL tool. The created ontology should be allowed be saved to an RDFStore of Apache Fuseki.