`owl2onto.cls`: Marking up OWL2 Ontologies in sTeX.*

Michael Kohlhase
Jacobs University, Bremen
`http://kwarc.info/kohlhase`

April 4, 2016

Abstract

The `owl2onto` class allows mark up OWL2 Ontologies in sTeX and generate OWL2-XML from them via the LaTeXML system.

Contents

*Version ? (last revised ?)

Experimental!
do not use!

1 The User Interface

The `owl2onto` package allows mark up ontology-based Metadata in LaTeX documents that can be harvested by automated tools or exported to PDF.[1]

EdN:1

The main idea behind the `owl2onto` class and package is to think of (documented) ontologies as documents, which present the knowledge behind ontology informally to the (human) reader and at the same time contain enough (hidden) information so that a formal ontology can be generated from them.

1.1 Package Options

`showmeta`

The `owl2onto` package takes a single option: `showmeta`. If this is set, then the metadata keys are shown (see [Koh15] for details and customization options).

1.2 Ontologies

`ontology`

The `owl2onto` class provides the `ontology` environment environment to declare OWL classes; it takes the place of the usual `document` environment[1], but its optional argument provides `KeyVal` attributes that allow to mark up the semantic aspects of the class, see Figure 3. In this example, we only have a single class declaration.

```
\documentclass{owl2onto}
\begin{ontology}[id=foaf,baseURI=http://xmlns.com/foaf/0.1/]
\declaration[type=Class,name=Agent,cseq=agent]
  {An agent (eg. person, group, software or physical artifact).}
\end{ontology}
```

Class: Agent
An agent (eg. person, group, software or physical artifact).

Example 1: A simple Ontology in LaTeX and its presentation

1.3 Declarations

`\declaration`

`type`

`name`

In general the `\declaration` macro can be used to declare the objects of various types in an ontology. The `type` key is used to specify this, its values range over the set[2]. The `name` attribute specifies the name of the declared object. This

EdN:2

[1] EdNote: continue

[1] Admittedly, it is somewhat unconventional to use the document environment for this, but this is the best way to ensure that we an OWL/XML document with a single document root.

[2] EdNote: give a list of all of them.

information is used in the XML generation; see Figure 2 for the result of generating
cseq XML from Figure 3. Finally, the `cseq` key allows to specify a command sequence for the object, which an be used in properties.

```
<?xml version="1.0"?>
<!--This OWL2 ontology is generated from an sTeX-encoded one via LaTeXML,
    you may want to reconsider editing it.-->
<Ontology xmlns="http://www.w3.org/2002/07/owl#"
         xmlns:stex="http://kwarc.info/ns/sTeX"
         xmlns:omdoc="http://omdoc.org/ns"
         ontologyIRI="http://xmlns.com/foaf/0.1/"
         xml:base="http://xmlns.com/foaf/0.1//"
         xml:id="foaf">
  <Import>owl2.omdoc#OWL2</Import>
  <Declaration stex:srcref="test.tex#textrange(from=5;0,to=5;64)">
    <Class IRI="Agent"/>
  </Declaration>
  <AnnotationAssertion>
    <AnnotationProperty IRI="http://www.w3.org/2000/01/rdf-schema#comment"/>
    <IRI>Agent</IRI>
    <Literal>An agent (eg. person, group, software or physical artifact).</Literal>
  </AnnotationAssertion>
</Ontology>
```

Example 2: The OWL/XML generated from Figure 3

1.4 Properties

\axiom The properties of the declared objects can be stated via the `\axiom` macro. Its
EdN:3 first argument is an OWL formula marked up in prefix notation[3], and the second one a natural language explanation.

```
\begin{document}
```

Axiom: $agent \| document$
Agents are not Documents

```
  <DisjointClasses>
    <Annotation>
      <AnnotationProperty IRI="http://www.w3.org/2000/01/rdf-schema"/>
      <Literal>Agents are not Documents</Literal>
    </Annotation>
    <Class IRI="Agent"/>
    <Class IRI="Document"/>
  </DisjointClasses>
```

Example 3: An Axiom

2 Limitations

In this section we document known limitations. If you want to help alleviate them, please feel free to contact the package author. Some of them are currently discussed in the sTeX GitHub repository [sTeX].

[3] EdNote: explain this better

1. none reported yet

3 The Implementation

The functionality is spread over the `owl2onto` class and package. The class provides the `document` environment and the `ontology` element corresponds to it, whereas the package provides the concrete functionality.

3.1 The `owl2onto` Class

We first define the `owl2onto` class, which on the LaTeX side just calls the article class.

```
⟨*cls⟩
\DeclareOption{showmeta}{\PassOptionsToPackage{\CurrentOption}{metakeys}}
\DeclareOption*{\PassOptionsToClass{\CurrentOption}{article}}
\ProcessOptions
\LoadClass{article}
\RequirePackage{modules}
\RequirePackage{owl2onto}
```

The `owl2onto` class also supplies the top-level `document` environment, which we redefined so that we can provide KeyVal arguments.

EdN:4

`document` The `document` environment is redefined to allow a baseURI[4]

```
\srefaddidkey{ontology}
\addmetakey{ontology}{baseURI}
\newcommand\ontology[1][]{\document\metasetkeys{ontology}{#1}%
\importmodule[owl2]{OWL2}%
\ifx\sref@id\@empty\begin{module}\else\begin{module}[id=\sref@id]\fi}
\newcommand\endontology{\end{module}\enddocument}
⟨/cls⟩
```

3.2 Classes and Properties

Before we provide the core functionality, we need to ensure that the `modules` and `presentation` packages are loaded. For LaTeXML we also initialize the package inclusions.

```
⟨*package⟩
\RequirePackage{amstext}
\RequirePackage{presentation}
```

We first ste up a utility macro that allows us to export values

`\exportvalue`

```
\def\exportvalue#1#2{%
\expandafter\def\csname\declaration@cseq #2\expandafter\endcsname\expandafter{#1}
```

[4] EdNote: @Deyan, need to remember the baseURI in a keyword, so that we can use it in the class and property environments

```
\expandafter\g@addto@macro\csname module@defs@\mod@id\expandafter\endcsname\expandafter%
{\expandafter\def\csname\declaration@cseq #2\expandafter\endcsname\expandafter{#1}}}
```

\declaration

```
\addmetakey{declaration}{id}
\addmetakey{declaration}{cseq}
\addmetakey{declaration}{type}
\addmetakey{declaration}{name}
\newcommand\declaration[2][]{\metasetkeys{declaration}{#1}%
\ifx\declaration@cseq\@empty\else
\expandafter\exportvalue{\ontology@baseURI\declaration@name}{@URI}
\expandafter\exportvalue{\expandafter\text\declaration@name}{}
\fi
\noindent\textbf{\declaration@type: \declaration@name}\par\noindent #2\par\noindent}
```

\axiom

```
\addmetakey{axiom}{id}
\addmetakey{axiom}{cseq}
\addmetakey{axiom}{type}
\addmetakey{axiom}{name}
\newcommand\axiom[3][]{\metasetkeys{axiom}{#1}%
\noindent\textbf{Axiom:} #2\par\noindent #3\par\noindent}
```

3.3 Using Ontologies

useontology

```
\def\useontology#1#2{\input}
```

⟨/package⟩

References

[Koh15] Michael Kohlhase. *metakeys.sty: A generic framework for extensible Metadata in LaTeX*. Tech. rep. Comprehensive TeX Archive Network (CTAN), 2015. URL: http://www.ctan.org/tex-archive/macros/latex/contrib/stex/metakeys/metakeys.pdf.

[sTeX] *KWARC/sTeX*. URL: https://svn.kwarc.info/repos/stex (visited on 05/15/2015).