smglom.cls/sty: Semantic Multilingual Glossary for Math

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

May 16, 2014

Abstract

The smglom package is part of the sTEX collection, a version of TEX/LATEX that allows to markup TEX/LATEX documents semantically without leaving the document format, essentially turning TEX/LATEX into a document format for mathematical knowledge management (MKM).

This package supplies an infrastructure for writing OMDoc glossary entries.

Contents

1 Introduction **2**

2 The User Interface **2**
2.1 Package and Class Options 2

3 Implementation: The SMGloM Class **3**
3.1 Class Options 3
3.2 For Module Definitions 4
3.3 For Language Bindings 5

1 Introduction

2 The User Interface

2.1 Package and Class Options

`smglom.cls` accepts all options of the `omdoc.cls` and `article.cls` and just passes them on to these.

3 Implementation: The SMGloM Class

3.1 Class Options

To initialize the `smglom` class, we pass on all options to `omdoc.cls`

```
⟨*cls⟩
\DeclareOption*{\PassOptionsToClass{\CurrentOption}{omdoc}}
\ProcessOptions
⟨/cls⟩
⟨*ltxml.cls | ltxml.sty⟩
# -*- CPERL -*-
package LaTeXML::Package::Pool;
use strict;
use warnings;
use LaTeXML::Package;

DeclareOption(undef,sub {PassOptions('article','cls',ToString(Digest(T_CS('\CurrentOption'))));
ProcessOptions();
⟨/ltxml.cls | ltxml.sty⟩
```

We load `omdoc.cls`, and the desired packages. For the LaTeXML bindings, we make sure the right packages are loaded.

```
⟨*cls⟩
\LoadClass{omdoc}
\RequirePackage{smglom}
⟨/cls⟩
⟨*sty⟩
\RequirePackage{amstext}
\RequirePackage{modules}
\RequirePackage{dcm}
\RequirePackage{statements}
\RequirePackage{sproof}
\RequirePackage{cmath}
\RequirePackage{presentation}
\RequirePackage{amsfonts}
⟨/sty⟩
⟨*ltxml.cls⟩
LoadClass('omdoc');
RequirePackage('smglom');
⟨/ltxml.cls⟩
⟨*ltxml.sty⟩
RequirePackage('amstext');
RequirePackage('modules');
RequirePackage('dcm');
RequirePackage('statements');
RequirePackage('sproof');
RequirePackage('cmath');
RequirePackage('presentation');
RequirePackage('amsfonts');
⟨/ltxml.sty⟩
```

3.2 For Module Definitions

`gimport` just a shortcut

```
⟨ltxml.sty⟩RawTeX('
⟨*sty | ltxml.sty⟩
\newcommand\gimport[2][]{\def\@test{#1}%
\edef\mh@@repos{\mh@currentrepos}%
\ifx\@test\@empty\importmhmodule[repos=\mh@@repos,ext=tex,path=#2]{#2}%
\else\importmhmodule[repos=#1,ext=tex,path=#2]{#2}\fi
\mhcurrentrepos\mh@@repos\ignorespaces}
```

`guse` just a shortcut

```
\newcommand\guse[2][]{\def\@test{#1}%
\edef\mh@@repos{\mh@currentrepos}%
\ifx\@test\@empty\usemhmodule[repos=\mh@@repos,ext=tex,path=#2]{#2}%
\else\usemhmodule[repos=#1,ext=tex,path=#2]{#2}\fi
\mhcurrentrepos\mh@@repos\ignorespaces}
```

`gadopt` just a shortcut

```
\newcommand\gadopt[2][]{\def\@test{#1}%
\edef\mh@@repos{\mh@currentrepos}%
\ifx\@test\@empty\adoptmhmodule[repos=\mh@@repos,ext=tex,path=#2]{#2}%
\else\adoptmhmodule[repos=#1,ext=tex,path=#2]{#2}\fi
\mhcurrentrepos\mh@@repos\ignorespaces}
⟨/sty | ltxml.sty⟩
⟨ltxml.sty⟩');
```

`*nym`

```
⟨*sty⟩
\newcommand\hypernym[3][]{\if@importing\else\par\noindent #2 is a hypernym of #3\fi}
\newcommand\hyponym[3][]{\if@importing\else\par\noindent #2 is a hyponym of  #3\fi}
\newcommand\meronym[3][]{\if@importing\else\par\noindent #2 is a meronym of #3\fi}
⟨/sty⟩
⟨*ltxml.sty⟩
DefConstructor('\hypernym [] {}{}',"");
DefConstructor('\hyponym [] {}{}',"");
DefConstructor('\meronym [] {}{}',"");
⟨/ltxml.sty⟩
```

EdN:1 `\MSC` to define the Math Subject Classification, [1]

```
⟨*sty⟩
\newcommand\MSC[1]{\if@importing\else MSC: #1\fi}
⟨/sty⟩
⟨*ltxml.sty⟩
DefConstructor('\MSC{}',"");
⟨/ltxml.sty⟩
```

[1] EdNote: MK: what to do for the LaTeXML side?

3.3 For Language Bindings

Here we adapt the `smultiling` functionality to the special situation, where the module and file names are identical by design.

`gviewsig` The `gviewsig` environment is just a layer over the `viewsig` environment with the keys suitably adapted.

```
⟨ltxml.sty⟩RawTeX('
⟨*sty | ltxml.sty⟩
\newenvironment{gviewsig}[4][]{\def\test{#1}\ifx\@test\@empty%
\begin{mhviewsig}[frompath=#3,topath=#4]{#2}{#3}{#4}\else
\begin{mhviewsig}[frompath=#3,topath=#4,#1]{#2}{#3}{#4}\fi}
{\end{mhviewsig}}
```

`gviewnl` The `gve` environment is just a layer over the `viewnl` environment with the keys suitably adapted.

```
\newenvironment{gviewnl}[5][]{\def\@test{#1}\ifx\@test\@empty%
\begin{mhviewnl}[frompath=#4,topath=#5]{#2}{#3}{#4}{#5}\else%
\begin{mhviewnl}[#1,frompath=#4,topath=#5]{#2}{#3}{#4}{#5}\fi
\smg@select@language{#3}}
{\end{mhviewnl}}
⟨/sty | ltxml.sty⟩
⟨ltxml.sty⟩');
```