`smglom.cls/sty`: Semantic Multilingual Glossary for Math

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

April 6, 2016

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 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` as well as the `stex` and `smglom` packages.

```
1 ⟨*cls⟩
2 \DeclareOption*{\PassOptionsToClass{\CurrentOption}{omdoc}
3                             \PassOptionsToPackage{\CurrentOption}{stex}
4                             \PassOptionsToPackage{\CurrentOption}{smglom}}
5 \ProcessOptions
```

We load `omdoc.cls`, the `smglom` package that provides the SMGloM-specific functionality[1], and the `stex` package to allow OMDoc compatibility.

```
6 \LoadClass{omdoc}
7 \RequirePackage{smglom}
8 \RequirePackage{stex}
9 \RequirePackage{amstext}
10 \RequirePackage{amsfonts}
11 ⟨/cls⟩
```

Now we do the same thing for the package; first the options, which we just pass on to the `stex` package.

```
12 ⟨*sty⟩
13 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{statements}
14                             \PassOptionsToPackage{\CurrentOption}{structview}
15                             \PassOptionsToPackage{\CurrentOption}{smultiling}}
16 \ProcessOptions
```

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

```
17 \RequirePackage{statements}
18 \RequirePackage[langfiles]{smultiling}
19 \RequirePackage{structview}
```

3.2 For Module Definitions

\gimport Just a shortcut, we have a starred and unstarred version, the first one is conservative. For example, if we execute:

```
\gimport[smglom/numberfields]{naturalnumbers}
```

First we are redirected to `\@gimport@nostar`, we store the `smglom/numberfields`⟨*the repo's path*⟩ in `\@test`, then store `\mh@currentrepos`⟨*current directory*⟩ in `\mh@repos`. If no repo's path is offered, that means the module to import is

---

[1]EDNOTE: MK:describe that above

under the same directory, so we let `repos=\mh@repos` and pass bunch of parameters to `\importmhmodule`, which is defined in `module.sty`. If there's a repo's path, then we let repos=⟨*the repo's path*⟩. Finally we use `\mhcurrentrepos`(defined in `module.sty`) to change the `\mh@currentrepos`.

```
20 \def\gimport{\@ifstar\@gimport@star\@gimport@nostar}%
21 \newrobustcmd\@gimport@star[2][]{%
22   \def\@test{#1}%
23   \edef\mh@@repos{\mh@currentrepos}%
24   \ifx\@test\@empty%
25     \importmhmodule[conservative,repos=\mh@@repos,ext=tex,path=#2]{#2}%
26   \else%
27     \importmhmodule[conservative,repos=#1,ext=tex,path=#2]{#2}%
28   \fi%
29   \mhcurrentrepos{\mh@@repos}%
30   \ignorespacesandpars%
31 }%
32 \newrobustcmd\@gimport@nostar[2][]{%
33   \def\@test{#1}%
34   \edef\mh@@repos{\mh@currentrepos}%
35   \ifx\@test\@empty%
36     \importmhmodule[repos=\mh@@repos,ext=tex,path=#2]{#2}%
37   \else%
38     \importmhmodule[repos=#1,ext=tex,path=#2]{#2}%
39   \fi%
40   \mhcurrentrepos{\mh@@repos}%
41   \ignorespacesandpars%
42 }%
```

guse   just a shortcut

```
43 \newrobustcmd\guse[2][]{%
44   \def\@test{#1}%
45   \edef\mh@@repos{\mh@currentrepos}%
46   \ifx\@test\@empty%
47     \usemhmodule[repos=\mh@@repos,ext=tex,path=#2]{#2}%
48   \else%
49     \usemhmodule[repos=#1,ext=tex,path=#2]{#2}%
50   \fi%
51   \mhcurrentrepos{\mh@@repos}%
52   \ignorespacesandpars%
53 }%
```

*nym

```
54 \newrobustcmd\hypernym[3][]{\if@importing\else\par\noindent #2 is a hypernym of #3\fi}%
55 \newrobustcmd\hyponym[3][]{\if@importing\else\par\noindent #2 is a hyponym of  #3\fi}%
56 \newrobustcmd\meronym[3][]{\if@importing\else\par\noindent #2 is a meronym of #3\fi}%
```

EdN:2   \MSC   to define the Math Subject Classification, [2]

```
57 \newrobustcmd\MSC[1]{\if@importing\else MSC: #1\fi\ignorespacesandpars}%
```

---

[2]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 `mhviewsig` environment with the keys suitably adapted.

```
58 \newenvironment{gviewsig}[4][]{%
59   \def\test{#1}%
60   \ifx\@test\@empty%
61     \begin{mhviewsig}[frompath=#3,topath=#4]{#2}{#3}{#4}%
62   \else%
63     \begin{mhviewsig}[frompath=#3,topath=#4,#1]{#2}{#3}{#4}%
64   \fi%
65   \ignorespacesandpars%
66 }{%
67   \end{mhviewsig}%
68   \ignorespacesandparsafterend%
69 }%
```

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

```
70 \newenvironment{gviewnl}[5][]{%
71   \def\@test{#1}\ifx\@test\@empty%
72     \begin{mhviewnl}[frompath=#4,topath=#5]{#2}{#3}{#4}{#5}%
73   \else%
74     \begin{mhviewnl}[frompath=#4,topath=#5,#1]{#2}{#3}{#4}{#5}%
75   \fi%
76 \ignorespacesandpars%
77 }{%
78   \end{mhviewnl}%
79   \ignorespacesandparsafterend%
80 }%
```

EdN:3   \gincludeview   [3]

```
81 \newcommand\gincludeview[2][]{\ignorespacesandpars}%
```

3.4   Authoring States

We add a key to the module environment.

```
82 \addmetakey{module}{state}%
```

3.5   Shadowing of repositories

\repos@macro   \repos@macro parses a GitLab repository name ⟨*group*⟩/⟨*name*⟩ and creates an internal macro name from that, which will be used

```
83 \def\repos@macro#1/#2;{#1@shadows@#2}%
```

---

[3]EDNOTE: This is fake for now, needs to be implemented and documented

\shadow \shadow{⟨*orig*⟩}{⟨*fork*⟩} declares a that the private repository ⟨*fork*⟩ shadows the MathHub repository ⟨*orig*⟩. Internally, it simply defines an internal macro with the shadowing information.

```
84 \def\shadow#1#2{\@namedef{\repos@macro#1;}{#2}}%
```

\MathHubPath \MathHubPath{⟨*repos*⟩} computes the path of the fork that shadows the MathHub repository ⟨*repos*⟩ according to the current \shadow specification. The computed path can be used for loading modules from the private version of ⟨*repos*⟩.

```
85 \def\MathHubPath#1{\@ifundefined{\repos@macro#1;}{#1}{\@nameuse{\repos@macro#1;}}}%
86 ⟨/sty⟩
```