

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

Michael Kohlhasse  
FAU Erlangen-Nürnberg  
<http://kwarc.info/kohlhasse>

November 23, 2019

## Abstract

The `smglom` package and class are part of the  $\text{\LaTeX}$  collection, a version of  $\text{\TeX}/\text{\LaTeX}$  that allows to markup  $\text{\TeX}/\text{\LaTeX}$  documents semantically without leaving the document format, essentially turning  $\text{\TeX}/\text{\LaTeX}$  into a document format for mathematical knowledge management (MKM).

This package supplies an infrastructure for writing OMDoc glossary entries.

## Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>The User Interface</b>	<b>2</b>
2.1	Package and Class Options . . . . .	2
2.2	Convenience Macros for SMGloM Modules . . . . .	2
2.3	Terminological Relations . . . . .	2
2.4	Namespaces and Alignments . . . . .	2
2.5	Presenting Glossaries . . . . .	3
<b>3</b>	<b>Implementation: The SMGloM Class</b>	<b>5</b>
3.1	Class Options . . . . .	5
3.2	Convenience Macros for SMGloM Modules . . . . .	5
3.3	Terminological Relations . . . . .	7
3.4	Namespaces and Alignments . . . . .	7
3.5	For Language Bindings . . . . .	7
3.6	Authoring States, etc . . . . .	8
3.7	Shadowing of repositories . . . . .	8
3.8	Building Glossaries . . . . .	9

# 1 Introduction

We use  $\text{\texttt{S}\TeX}$  as the surface language for the SMGLoM (Semantic Multilingual Glossary of Mathematics), see [Gin+16; Koh14; SMG]. The `smglom` package and class provides some infrastructure to make this more convenient.

## 2 The User Interface

The `smglom` package provides convenience macros on top of the  $\text{\texttt{S}\TeX}$  infrastructure to simplify writing SMGLoM glossary modules and make them more concise for reading. The `smglom` class just sets up the necessary  $\text{\texttt{S}\TeX}$  packages and loads the `smglom` package.

### 2.1 Package and Class Options

`smglom.sty` accepts all options of the  $\text{\texttt{S}\TeX}$  package and passes them along to `stex.sty` [Koh18]. `smglom.cls` also does that for the classes `omdoc.cls` [Kohlhase:smomdl] and `article.cls`.

### 2.2 Convenience Macros for SMGLoM Modules

The SMGLoM source files are more regular than arbitrary  $\text{\texttt{S}\TeX}$  files. In particular,

- make heavy use of the `smultiling` package for multilingual  $\text{\texttt{S}\TeX}$ ,
- use the `mathhub` extensions to  $\text{\texttt{S}\TeX}$  for file system organization,
- enforce the one-module-one-file convention and make sure that the module name must be the same as the (base name) of the file.

This allows use to abbreviate e.g.

```
\importmhmodule[mhrepos=lib/archive,path=current/modfile]{modname}
```

```
\gimport by\gimport[lib/archive]{modname} and analogously for \guse. 1
\guse
```

### 2.3 Terminological Relations

<sup>2</sup>

### 2.4 Namespaces and Alignments

```
\symalign
```

<sup>3</sup> In SMGLoM, we often want to align the content of glossary modules to formalizations, e.g. to take advantage of type declarations there. The `\symalign` macro takes two regular arguments: the first is the name symbol declared in the current module (e.g. by a `\syymi`), and the second the URI name of a symbol in an external theory in the form  $\langle theory \rangle ? \langle name \rangle$ .

---

<sup>1</sup>EDNOTE: document them

<sup>2</sup>EDNOTE: document them

<sup>3</sup>EDNOTE: MK: maybe this should go into some other module; it seems awfully foundational.

As full MMT URIs are of the form  $\langle URI \rangle ? \langle theory \rangle ? \langle name \rangle$ , we need a way to specify the  $\langle URI \rangle$ . We adopt the system of **namespaces** of in MMT: the `\namespace` macro declares a namespace URI. If the optional argument is given, then this is a namespace abbreviation declaration, which can be used later, for instance in `\symalign` that takes an optional first argument: the namespace of the external theory.

The situation below is typical. We first declare the namespace abbreviation `sets` and then use the `\modalign` macro to specify that the external theory `sets:?ESet` is the default alignment target, i.e. any symbol that in the local `emptyset` module is aligned by default to the symbol with the same name in the external `sets:?ESet` theory.

```
\begin{modsig}[creators=miko]{emptyset}
  \gimport{set}
  \namespace[sets]{http://mathhub.info/MitM/smgglom/sets}
  \modalign[sets]{ESet}

  \symdef{eset}{\emptyset}
  \symi{non-empty}
  \symalign{non-empty}{ESet?non_empty}
\end{modsig}
```

The default alignment breaks down for the symbol `non-empty`, so we specify an alignment to the symbol `Eset?non_empty` via `\symalign`.

## 2.5 Presenting Glossaries

`smglossary` The `smglom` package provides the `smglossary` environment for presenting glossaries. This expects a sequence of

- `smentry`
  - glossary entries marked up using the `smentry` environment, which contains a definition.
- `\smsynonymref`
  - synonym references marked up `\smsynonymref`
- `\smjointdefref`
  - joint definition references marked up `\smjointdefref`

The latter two mark up cross references for definitions that contain more than one `\defi*` and would otherwise result in multiple (often more than a handful) copies of the same definition and thus lead to rambling glossaries.

The following snippet is a typical example, showing all three cases.

```
\begin{smglossary}
  \smjointdefref{zero vector}{x6e12a4211dd6546c}{vector space}
  \begin{smentry}{\hypertarget{x4d4e8afd0e133715}{zerofree}}{smglom/numbers}
    \guse[smglom/numbers]{zerofree}
    An \trefi[integernumbers]{integer} whose decimal digits
    \trefi[positional-number-system]{digit} no zeros is said to be \defi{zerofree}.
  \end{smentry}
\end{smglossary}
```

```

\end{smentry}
\smsynonymref{well-ordering}{x1e9bbb88fb4d90b3}{well-order}
\end{smglossary}

```

The  $\text{\TeX}$  universe has a set of LMH scripts [[URL:lmhtools:github](https://github.com/TeX-LMH)] that allow to generate glossaries and dictionaries from  $\text{\TeX}$  sources, such as [SMG].

## 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<sup>4</sup>, 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}{dcm}
15                                     \PassOptionsToPackage{\CurrentOption}{cmath}
16                                     \PassOptionsToPackage{\CurrentOption}{structview}
17                                     \PassOptionsToPackage{\CurrentOption}{smutiling}}
18 \ProcessOptions
```

We load `omdoc.cls`, and the desired packages. For the L<sup>A</sup>T<sub>E</sub>XML bindings, we make sure the right packages are loaded.

```
19 \RequirePackage{statements}
20 \RequirePackage[langfiles]{smutiling}
21 \RequirePackage{structview}
22 \RequirePackage{dcm}
23 \RequirePackage{cmath}
24 \RequirePackage[base]{babel}
```

We set up triggers for the other languages, currently only German.

```
25 \AfterBabelLanguage{ngerman}{\input{smglom-ngerman.ldf}}
```

### 3.2 Convenience Macros for SMGloM Modules

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

---

<sup>4</sup>EdNOTE: MK:describe that above

`\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 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`.

```

26 \def\gimport{\@ifstar\@gimport@star\@gimport@nostar}%
27 \newrobustcmd\@gimport@star[2] [] {%
28   \def\@test{#1}%
29   \edef\mh@currentrepos{\mh@currentrepos}%
30   \ifx\@test\@empty%
31     \importmhmodule[conservative, repos=\mh@currentrepos, ext=tex, path=#2]{#2}%
32   \else%
33     \importmhmodule[conservative, repos=#1, ext=tex, path=#2]{#2}%
34   \fi%
35   \mhcurrentrepos{\mh@currentrepos}%
36   \ignorespacesandpars%
37 }%
38 \newrobustcmd\@gimport@nostar[2] [] {%
39   \def\@test{#1}%
40   \edef\mh@currentrepos{\mh@currentrepos}%
41   \ifx\@test\@empty%
42     \importmhmodule[repos=\mh@currentrepos, ext=tex, path=#2]{#2}%
43   \else%
44     \importmhmodule[repos=#1, ext=tex, path=#2]{#2}%
45   \fi%
46   \mhcurrentrepos{\mh@currentrepos}%
47   \ignorespacesandpars%
48 }%

```

**guse** just a shortcut

```

49 \newrobustcmd\guse[2] [] {\def\@test{#1}%
50   \edef\mh@currentrepos{\mh@currentrepos}%
51   \ifx\@test\@empty%
52     \usemhmodule[repos=\mh@currentrepos, ext=tex, path=#2]{#2}%
53   \else%
54     \usemhmodule[repos=#1, ext=tex, path=#2]{#2}%
55   \fi%
56   \mhcurrentrepos{\mh@currentrepos}%
57   \ignorespacesandpars%
58 }%

```

**gstructure** we essentially copy over the definition of `mhstructure`, but adapt it to the SM-GloM situation.

```

59 \newenvironment{gstructure}[3] [] {\def\@test{#1}%

```

```

60 \xdef\mh@@@repos{\mh@currentrepos}%
61 \ifx\@test\@empty%
62   \gdef\@doit{\importmhmodule[repos=\mh@@@repos,path=#3,ext=tex]{#3}}%
63 \else%
64   \gdef\@doit{\importmhmodule[repos=#1,path=#3,ext=tex]{#3}}%
65 \fi%
66 \ifmod@show\par\noindent structure import "#2" from module #3 \@doit\fi%
67 \ignorespacesandpars}
68 {\aftergroup\@doit\ifmod@show end import\fi%
69 \ignorespacesandparsafterend}

```

### 3.3 Terminological Relations

\*nym

```

70 \newrobustcmd\hypernym[3] [] {\if@importing\else\par\noindent #2 is a hypernym of #3\fi}%
71 \newrobustcmd\hyponym[3] [] {\if@importing\else\par\noindent #2 is a hyponym of #3\fi}%
72 \newrobustcmd\meronym[3] [] {\if@importing\else\par\noindent #2 is a meronym of #3\fi}%

```

EdN:5

\MSC to define the Math Subject Classification,<sup>5</sup>

```

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

```

### 3.4 Namespaces and Alignments

\namespace

```

74 \newcommand\namespace[2] [] {\ignorespaces}

```

\modalign

```

75 \newcommand\modalign[2] [] {\ignorespaces}

```

\symalign

```

76 \newcommand\symalign[3] [] {\ignorespaces}

```

### 3.5 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.

```

77 \newenvironment{gviewsig}[4] [] {% keys, id, from, to
78   \def\test{#1}%
79   \ifx\@test\@empty%
80     \begin{mhviewsig}[frompath=#3,topath=#4]{#2}{#3}{#4}%
81   \else%
82     \begin{mhviewsig}[frompath=#3,topath=#4,#1]{#2}{#3}{#4}%
83   \fi%

```

---

<sup>5</sup>EdNOTE: MK: what to do for the LaTeXML side?

```

84 \ignorespacesandpars%
85 }{%
86 \end{mhviewsig}%
87 \ignorespacesandparsafterend%
88 }%

```

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

```

89 \newenvironment{gviewnl}[5][{}]{% keys, id, lang, from, to
90 \def\@test{#1}\ifx\@test\@empty%
91 \begin{mhviewnl}[frompath=#4,topath=#5]{#2}{#3}{#4}{#5}%
92 \else%
93 \begin{mhviewnl}[frompath=#4,topath=#5,#1]{#2}{#3}{#4}{#5}%
94 \fi%
95 \ignorespacesandpars%
96 }{%
97 \end{mhviewnl}%
98 \ignorespacesandparsafterend%
99 }%

```

EdN:6

```

\gincludeview 6
100 \newcommand\gincludeview[2][{}]{\ignorespacesandpars}%

```

### 3.6 Authoring States, etc

We add a key to the module environment.

```

101 \addmetakey{module}{state}%

```

### 3.7 Shadowing of repositories

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

```

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

```

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

```

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

```

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

```

104 \def\MathHubPath#1{\@ifundefined{\repos@macro#1;}{#1}{\@nameuse{\repos@macro#1;}}}%

```

---

<sup>6</sup>EDNOTE: This is fake for now, needs to be implemented and documented



### 3.8 Building Glossaries

```
smentry
105 \newenvironment{smentry}[2]%
106 {\item[\textbf{#1}]\mhcurrentrepos{#2}\begin{module}[id=foo]\begin{definition}[display=flow]}
107 {\end{definition}\end{module}}

smglossary
108 \newenvironment{smglossary}{\begin{itemize}}{\end{itemize}}

\smsynonymref
109 \newcommand\smglom@synonymref@kw{See}
110 \newcommand\smsynonymref[3]{\item[\textbf{#1}] \smglom@synonymref@kw\ \textcolor{blue}{\hyperlin

\smjointdefref
111 \newcommand\smglom@jointref@kw{Defined along with}
112 \newcommand\smjointdefref[3]{\item[\textbf{#1}] \smglom@jointref@kw\ \textcolor{blue}{\hyperlin
113 \</sty>
```

## References

- [Gin+16] Deyan Ginev et al. “The SMGloM Project and System. Towards a Terminology and Ontology for Mathematics”. In: *Mathematical Software - ICMS 2016 - 5th International Congress*. Ed. by Gert-Martin Greuel et al. Vol. 9725. LNCS. Springer, 2016. DOI: 10.1007/978-3-319-42432-3. URL: <http://kwarc.info/kohlhase/papers/icms16-smglom.pdf>.
- [Koh14] Michael Kohlase. “A Data Model and Encoding for a Semantic, Multilingual Terminology of Mathematics”. In: *Intelligent Computer Mathematics*. Conferences on Intelligent Computer Mathematics (Coimbra, Portugal, July 7–11, 2014). Ed. by Stephan Watt et al. LNCS 8543. Springer, 2014, pp. 169–183. ISBN: 978-3-319-08433-6. URL: <http://kwarc.info/kohlhase/papers/cicm14-smglom.pdf>.
- [Koh18] Michael Kohlase. *sTeX: Semantic Markup in T<sub>E</sub>X/L<sup>A</sup>T<sub>E</sub>X*. Tech. rep. 2018. URL: <https://github.com/sLaTeX/sTeX/raw/master/sty/stex.pdf>.
- [SMG] *SMGloM Git Repository*. URL: <http://gl.mathhub.info/smgloM/smgloM> (visited on 07/10/2013).