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

Michael Kohlhase
FAU Erlangen-Nürnberg
http://kwarc.info/kohlhase

November 23, 2019

Abstract

The `smglom` package and class are 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

We use STEX 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 STEX infrastructure to simplify writing SMGLoM glossary modules and make them more concise for reading. The `smglom` class just sets up the necessary STEX packages and loads the `smglom` package.

2.1 Package and Class Options

`smglom.sty` accepts all options of the STEX package and passes them along to `stex.sty` [Koh18]. `smglom.cls` also does that for the casses `omdoc.cls` [**Kohlhase:smomdl**] and `article.cls`.

2.2 Convenience Macros for SMGloM Modules

The SMGLoM source files are more regular than arbitrary STEX files. In particular,
- make heavy use of the `smultiling` package for multilingual STEX,
- use the `mathhub` extensions to STEX 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`
`\guse`
by `\gimport[lib/archive]{modname}` and analogously for `\guse`. [1]

2.3 Terminological Relations

[2]

2.4 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`

---

[1] EDNOTE: document them
[2] EDNOTE: document them

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

The STEX universe has a set of LMH scripts [] that allow to generate glossaries and dictionaries from STEX 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[3], 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. But we also make sure that the modules package is loaded with the mh option, since the smglom package depends on these extensions.

```
12 ⟨∗sty⟩
13 \PassOptionsToPackage{mh}{modules}
14 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{statements}
15                                 \PassOptionsToPackage{\CurrentOption}{dcm}
16                                 \PassOptionsToPackage{\CurrentOption}{cmath}
17                                 \PassOptionsToPackage{\CurrentOption}{structview}
18                                 \PassOptionsToPackage{\CurrentOption}{smultiling}}
19 \ProcessOptions
```

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

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

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

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

---

[3]EDNOTE: MK:describe that above

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:

```
\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**.

```
27 \def\gimport{\@ifstar\@gimport@star\@gimport@nostar}%
28 \newrobustcmd\@gimport@star[2][]{%
29   \def\@test{#1}%
30   \edef\mh@@repos{\mh@currentrepos}%
31   \ifx\@test\@empty%
32     \importmhmodule[conservative,repos=\mh@@repos,ext=tex,path=#2]{#2}%
33   \else%
34     \importmhmodule[conservative,repos=#1,ext=tex,path=#2]{#2}%
35   \fi%
36   \mhcurrentrepos{\mh@@repos}%
37   \ignorespacesandpars%
38 }%
39 \newrobustcmd\@gimport@nostar[2][]{%
40   \def\@test{#1}%
41   \edef\mh@@repos{\mh@currentrepos}%
42   \ifx\@test\@empty%
43     \importmhmodule[repos=\mh@@repos,ext=tex,path=#2]{#2}%
44   \else%
45     \importmhmodule[repos=#1,ext=tex,path=#2]{#2}%
46   \fi%
47   \mhcurrentrepos{\mh@@repos}%
48   \ignorespacesandpars%
49 }%
```

guse just a shortcut

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

```
59 }%
```

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

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

3.3   Terminological Relations

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

EdN:4   \MSC   to define the Math Subject Classification, [4]

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

3.4   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.

```
75 \newenvironment{gviewsig}[4][]{% keys, id, from, to
76   \def\test{#1}%
77   \ifx\@test\@empty%
78     \begin{mhviewsig}[frompath=#3,topath=#4]{#2}{#3}{#4}%
79   \else%
80     \begin{mhviewsig}[frompath=#3,topath=#4,#1]{#2}{#3}{#4}%
81   \fi%
82   \ignorespacesandpars%
83 }{%
84   \end{mhviewsig}%
85   \ignorespacesandparsafterend%
86 }%
```

---

[4] EDNOTE: MK: what to do for the LaTeXML side?

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

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

\gincludeview [5]

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

3.5 Authoring States, etc

We add a key to the module environment.

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

3.6 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

```
100 \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.

```
101 \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 $\langle repos \rangle$.

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

3.7 Building Glossaries

smentry

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

---

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

smglossary

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

\smsynonymref

107 \newcommand\smglom@synonymref@kw{See}
108 \newcommand\smsynonymref[3]{\item[\textbf{#1}] \smglom@synonymref@kw\ \textcolor{blue}{\hyperli

\smjointdefref

109 \newcommand\smglom@jointref@kw{Defined along with}
110 \newcommand\smjointdefref[3]{\item[\textbf{#1}] \smglom@jointref@kw\ \textcolor{blue}{\hyperlin
111 ⟨/sty⟩

References

[]   *LMH Scripts*. GitHub repository at `https://github.com/sLaTeX/lmhtools`. accessed 15. 11. 2019. URL: `https://github.com/sLaTeX/lmhtools`.

[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 Kohlhase. "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 Kohlhase. *sTeX: Semantic Markup in TEX/LATEX*. Tech. rep. 2018. URL: `https://github.com/sLaTeX/sTeX/raw/master/sty/stex.pdf`.

[SMG]   *SMGloM: A Semantic Multilingual Glossary Resource for Mathematics*. URL: `http://gl.mathhub.info/smglom` (visited on 07/10/2019).