# MathHub Support for STEX\*

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

November 10, 2015

#### Abstract

The sref 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).

The  ${\tt mathhub}$  packages extend SIEX with support for the MathHub.info portal

### Contents

1	Intr	roduction	3
<b>2</b>	The	e User Interface	3
	2.1	Package Options	3
	2.2	modules-mh: MH Variants for Modules	3
	2.3	omtext-mh: MH Variants for OMText	4
	2.4	smultiling-mh: MH Variants for Multilinguality	4
	2.5	structview-mh: MH Variants for Structures and Views	4
	2.6	mikoslides-mh: Support for MiKo Slides	4
	2.7	problem-mh: Support for Problems	5
	2.8	hwexam-mh: Support for Assignments	5
3	Lim	itations	5
4	Imp	plementation	6
	4.1	General Infrastructure	6
	4.2	modules-mh: MH Variants for Modules	7
	4.3	omtext-mh: MH Variants for OMText	10
	4.4	smultiling-mh: MH Variants for Multilinguality	11
	4.5	structview-mh: MH Variants for Structures and Views	13

<sup>\*</sup>Version v1.0 (last revised 2015/11/04)

4.6	mikoslides-mh: Support for MiKo Slides	16
4.7	problem-mh: Support for Problems	16
4.8	hwexam-mh: Support for Assignments	17
4.9	Finale	19

#### 1 Introduction

Much of the STEX content is hosted on MathHub (http://MathHub.info), a portal and archive for flexiformal mathematics. MathHub offers GIT repositories (public and private escrow) for mathematical documentation projects, online and offline authoring and document development infrastructure, and a rich, interactive reading interface. The modules package supports repository-sensitive operations on MathHub.

Note that MathHub has two-level repository names of the form  $\langle group \rangle / \langle repo \rangle$ , where  $\langle group \rangle$  is a MathHub-unique repository group and  $\langle repo \rangle$  a repository name that is  $\langle group \rangle$ -unique. The file and directory structure of a repository is arbitrary – except that it starts with the directory source because they are Math Archives in the sense of [Hor+11]. But this structure can be hidden from the STEX author with MathHub-enabled versions of the STEX macros, which are defined in this package.

Caveat if you want to use the MathHub support macros (let's call them mh-variants), then every time a module is imported or a document fragment is included from another repos, the mh-variant \importmhmodule must be used, so that the "current repository" is set accordingly. To be exact, we only need to use mh-variants, if the imported module or included document fragment use mh-variants.

### 2 The User Interface

### 2.1 Package Options

none so far

#### 2.2 modules-mh: MH Variants for Modules

\importmhmodule

The importmhmodule macro is a variant of \importmodule with repository support. Instead of writing

\defpath{MathHub}{/user/foo/lmh/MathHub}
\importmodule[load=\MathHub{fooMH/bar/source/baz/foobar}]{foobar}

we can simply write (assuming that \MathHub is defined as above)

\importmhmodule[repos=fooMH/bar,path=baz/foobar]{foobar}

Note that the **\importmhmodule** form is more semantic, which allows more advanced document management features in MathHub.

If baz/foobar is the "current module", i.e. if we are on the MathHub path ...MathHub/fooMH/bar..., then stating the repository in the first optional argument is redundant, so we can just use

\importmhmodule[path=baz/foobar]{foobar}

if no file needs to loaded, \importmhmodule is the same as \importmodule.

\mhcurrentrepos

Of course, neither LATEX nor LATEXMLknow about the repositories when they are called from a file system, so we can use the \mhcurrentrepos macro to tell them. But this is only needed to initialize the infrastructure in the driver file. In particular, we do not need to set it in in each module, since the \importmhmodule macro sets the current repository automatically.

\usemhmodule \mhinputref \mhinput The \usemhmodule is the analog to \usemodule.

For this, the modules package supplies the mh-variants \mhinputref and \mhinput of the \inputref macro introduced above and normal LATEX \input macro.

#### 2.3 omtext-mh: MH Variants for OMText

\mhcgraphics

The \mhcgraphics macro is a variant of \mycgraphics with repository support. Instead of writing

\defpath{MathHub}{/user/foo/lmh/MathHub}
\mycgraphics{\MathHub{fooMH/bar/source/baz/foobar}}

we can simply write (assuming that \MathHub is defined as above)

\mhcgraphics[fooMH/bar]{baz/foobar}

Note that the \mhcgraphics form is more semantic, which allows more advanced document management features in MathHub.

#### 2.4 smultiling-mh: MH Variants for Multilinguality

1 2

#### 2.5 structview-mh: MH Variants for Structures and Views

3

#### 2.6 mikoslides-mh: Support for MiKo Slides

\mhframeimage

EdN:3

EdN:3

The \mhframeimage macro is a variant of \frameimage with repository support. Instead of writing

\defpath{MathHub}{/user/foo/lmh/MathHub}
\frameimage{\MathHub{fooMH/bar/source/baz/foobar}}

 $<sup>^{1}\</sup>mathrm{EdNote}$  needs to be documented

 $<sup>^2\</sup>mathrm{EdNote}$ : mhmodsig seems to be missing what happened?

 $<sup>^3\</sup>mathrm{EdNote}$ : needs to be documented

we can simply write (assuming that \MathHub is defined as above)

\mhframeimage[fooMH/bar]{baz/foobar}

Note that the \mhframeimage form is more semantic, which allows more advanced document management features in MathHub.

If baz/foobar is the "current module", i.e. if we are on the MathHub path ...MathHub/fooMH/bar..., then stating the repository in the first optional argument is redundant, so we can just use

\mhframeimage{baz/foobar}

#### 2.7 **problem-mh**: Support for Problems

\includemhproblem

The \includemhproblem macro is a variant of \importmodule with repository support. Instead of writing

\defpath{MathHub}{/user/foo/lmh/MathHub}
\includeproblem[pts=7]{\MathHub{fooMH/bar/source/baz/foobar}}

we can simply write (assuming that \MathHub is defined as above)

\includemhproblem[fooMH/bar]{baz/foobar}

Note that the \importmhproblem form is more semantic, which allows more advanced document management features in MathHub.

#### 2.8 hwexam-mh: Support for Assignments

\includemhassignment

The \includemhassignment macro is a variant of \includeassignment with repository support. Instead of writing

\defpath{MathHub}{/user/foo/lmh/MathHub}
\includeassignment[pts=7]{\MathHub{fooMH/bar/source/baz/foobar}}

we can simply write (assuming that \MathHub is defined as above)

\includemhassignment[fooMH/bar]{baz/foobar}

#### 3 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 ST<sub>E</sub>X GitHub repository [sTeX].

1. none reported yet.

### 4 Implementation

The sref package generates two files: the LATEX package (all the code between  $\langle *package \rangle$  and  $\langle /package \rangle$ ) and the LATEXML bindings (between  $\langle *ltxml \rangle$ ) and  $\langle /ltxml \rangle$ ). We keep the corresponding code fragments together, since the documentation applies to both of them and to prevent them from getting out of sync.

We first set up header information for the LATEXML binding files an the base package.

```
1 (*Itxml | modules.Itxml | omtext.Itxml | smultiling.Itxml | mikoslides.Itxml | problem.Itxml | hwexam.Itxml)
2 package LaTeXML::Package::Pool;
3 use strict;
4 use LaTeXML::Package;
5 (/ltxml | modules.ltxml | omtext.ltxml | smultiling.ltxml | mikoslides.ltxml | problem.ltxml | hwexam.ltxml)
6 (package)\ProvidesPackage{mathhub}[2015/11/04 v1.0 sTeX Support for MathHub.info]
7 (*package)
8 \DeclareOption*{}
9 \ProcessOptions
10 (/package)
11 (*ltxml)
12 DeclareOption(undef, sub {});
13 ProcessOptions();
14 \langle /ltxml \rangle
   Then we need to set up the packages by requiring the metakeys pack-
age [Koh15] to be loaded (in the right version).
15 (*package)
16 \RequirePackage{keyval}
17 (/package)
18 (*ltxml)
19 RequirePackage('keyval');
20 (/ltxml)
```

#### 4.1 General Infrastructure

\mhcurrentrepos \@mhcurrentrepos \mhcurrentrepos is used to initialize the current repository. If the repos has changed, it writes a call to the internal macro \@mhcurrentrepos for the aux file and calls it. So that the \importmodule calls there work with the correct repos.

```
21 (*package)
22 \newcommand\mhcurrentrepos[1] {%
    \edef\@test{#1}%
23
24
    \ifx\@test\mh@currentrepos% if new dir = old dir
25
      \relax% no need to change
26
    \else%
27
      \protected@write\@auxout{}{\string\@mhcurrentrepos{#1}}%
28
    \@mhcurrentrepos{#1}% define mh@currentrepos
29
31 \newcommand\@mhcurrentrepos[1]{\edef\mh@currentrepos{#1}}%
```

```
32 (/package)
                 33 (*ltxml)
                 34 DefMacro('\mhcurrentrepos{}','\@mhcurrentrepos{#1}');
                 35 DefMacro('\@mhcurrentrepos{#1}','\def\mh@currentrepos{#1}\@mhcurrentrepos{#1}');
                 36 DefConstructor('\@@mhcurrentrepos{}','',
                       afterDigest => sub{ AssignValue('current_repos',ToString($_[1]->getArg(1)),'global'); } );
                 38 (/ltxml)#$
\libinput
                the \libinput macro inputs from the lib directory of the MathHub repository
                 or the meta-inf/lib repos of the group.
                 39 (ltxml)RaxTeX('
                 40 (*package | ltxml)
                 41 \def\modules@@first#1/#2;{#1}
                 \label{limits} 42 \end{libinput} [1] {\end{libinjut} [1] {\end{libinjut} (\end{libinjut} )} % $$ $ (\end{libinjut} [1] {\end{libinjut} (\end{libinjut} )} $$ $ (\end{libinjut} [1] {\end{libinjut} (\end{libinjut} )} $$ $ (\end{libinjut} [1] {\end{libinjut} )} $$ $ (\end{libinjut} ]} $$ $ (\end{libinjut} ]
                 43 \IfFileExists{\@libfile}{\input\@libfile}%
                 44 {\edef\@@group{\expandafter\modules@@first\mh@currentrepos;}
                 45 \edef\@inffile{\MathHub{\@@group/meta-inf/lib/#1}}
                 46 \IfFileExists{\@inffile}{\input{\@inffile}}%
                 47 {\PackageError{modules}
                       {Library file missing, cannot input #1\MessageBreak%
                 49
                           Both \@libfile.tex\MessageBreak and \@inffile.tex\MessageBreak do not exit}%
                       {Check whether the file name is correct}}}}
                 51 \langle /package \mid ltxml \rangle
                 52 \langle \mathsf{ltxml} \rangle,;
```

#### 4.2 modules-mh: MH Variants for Modules

We set up package options and pass them on to the modules package, which we also load.

```
53 \*modules\
54 \ProvidesPackage{modules-mh}[2015/11/04 v1.0 MathHub support for the sTeX modules package]
55 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{modules}}
56 \ProcessOptions
57 \RequirePackage{modules}
58 \RequirePackage{modules}
59 \sqrt{modules}
60 \left\{\text{modules.ltxml}\right\}
61 \DeclareOption(undef,sub{\PassOptions('modules','sty',ToString(Digest(T_CS('\CurrentOption'))));}
62 \ProcessOptions();
63 \RequirePackage('modules');
64 \RequirePackage('mathhub');
65 \sqrt{\modules.ltxml}
```

\importmhmodule

The \importmhmodule[ $\langle key=value\ list\rangle$ ] {module} saves the current value of \mh@currentrepos in a local macro \mh@Grepos, resets \mh@currentrepos to the new value if one is given in the optional argument, and after importing resets \mh@currentrepos to the old value in \mh@Grepos. We do all the \ifx comparison with an \expandafter, since the values may be passed on from other key bindings. Parameters will be passed to \importmodule.

```
66 (*modules)
67 \srefaddidkey{importmhmodule}%
68 addmetakey{importmhmodule}{repos}% saves the repo's path. E.g: smglom/numberfield
69 \addmetakey{importmhmodule}{path}% saves the module name. E.g: naturalnumbers
70 \addmetakey[sms]{importmhmodule}{ext}% saves the extension: E.g: tex
71 \addmetakey[false]{importmhmodule}{conservative}[true]%
72 \newcommand\importmhmodule[2][]{%
     \metasetkeys{importmhmodule}{#1}%
73
     \ifx\importmhmodule@path\@empty% if module name is not set
74
       \importmodule[ext=\importmhmodule@ext,id=\importmhmodule@id]{#2}%
75
     \else%
76
 77
       \edef\mh@@repos{\mh@currentrepos}% remember so that we can reset it.
       \ifx\importmhmodule@repos\@empty% if in the same repos
 78
         \relax% no need to change mh@currentrepos, i.e, current dirctory.
 79
       \else%
80
         \mhcurrentrepos{\importmhmodule@repos}% change it.
81
82
       %,{\importmodule[load=\MathHub{\mh@currentrepos/source/\importmhmodule@path},
83
       ext=\importmhmodule@ext,id=\importmhmodule@id]{#2}%
84
85
       \mhcurrentrepos{\mh@@repos}% after importing, reset to old value
86
     \ignorespaces%
87
88 }%
89 (/modules)
90 (*modules.ltxml)
91 DefKeyVal('importmhmodule', 'id', 'Semiverbatim');
92 DefKeyVal('importmhmodule', 'repos', 'Semiverbatim');
93 DefKeyVal('importmhmodule', 'path', 'Semiverbatim');
94 DefKeyVal('importmhmodule', 'ext', 'Semiverbatim');
95 DefKeyVal('importmhmodule', 'conservative', 'Semiverbatim');
96 DefConstructor('\importmhmodule OptionalKeyVals:importmhmodule {}',
97
          "<omdoc:imports "
98
          . "from='?%GetKeyVal(#1,'load')(&canonical_omdoc_path(&GetKeyVal(#1,'load')))()###2'"
                   . "?&defined(&GetKeyVal(#1,'conservative'))(load='&GetKeyVal(#1,'conservative')'
99
100
      afterDigest => \&importMHmoduleI);
101
102 sub importMHmoduleI {
     my ($stomach, $whatsit) = @_;
103
    my $keyval = $whatsit->getArg(1);
     my $id = $whatsit->getArg(2);
105
106
     if ($keyval) {
       my $repos = ToString($keyval->getValue('repos'));
107
       my $path = ToString($keyval->getValue('path'));
108
       my $current_repos = LookupValue('current_repos');
109
110
       if (!$repos) { # Use the implicit current repository
111
         $repos = $current_repos; }
112
       my $defpaths = LookupValue('defpath');
113
       my $load_path = ($$defpaths{MathHub}).$repos.'/source/'.$path;
       $keyval->setValue('load',$load_path);
114
       AssignValue('current_repos' => $repos, 'global');
115
```

```
importmoduleI($stomach,$whatsit);
             116
                     AssignValue('current_repos' => $current_repos, 'global'); }
             117
             118
                     importmoduleI($stomach,$whatsit); }
             119
                   return; }
             120
             121
             122 DefConstructor('\importMHmoduleI OptionalKeyVals:importmhmodule {}', '',
                    afterDigest=> \&importMHmoduleI );#$
             124 (/modules.ltxml)
              and now the analogs
\usemhmodule
             _{125} \; \langle * modules \rangle
             126 \newcommand\usemhmodule[2][]{%
             127
                   \metasetkeys{importmhmodule}{#1}%
             128
                   \ifx\importmhmodule@path\@empty%
             129
                     \usemodule[ext=\importmhmodule@ext,id=\importmhmodule@id]{#2}%
             130
                   \else%
                     \edef\mh@currentrepos}%
             131
             132
                     \ifx\importmhmodule@repos\@empty%
             133
             134
                       \mhcurrentrepos{\importmhmodule@repos}%
             135
             136
                     \usemodule[load=\MathHub{\mh@currentrepos/source/\importmhmodule@path},ext=\importmhmodule@
             137
                     \mhcurrentrepos\mh@@repos%
             138
                   \fi%
             139
                   \ignorespaces%
             140 }%
             141 (/modules)
             142 (*modules.ltxml)
             143 DefConstructor('\usemhmodule OptionalKeyVals:importmhmodule {}',
                    "<omdoc:uses from='?&GetKeyVal(#1,'load')(&canonical_omdoc_path(&GetKeyVal(#1,'load')))()###</pre>
             144
             145
                    afterDigest => \&importMHmoduleI);
             146 (/modules.ltxml)
\mhinputref
             147 (modules.ltxml)RawTeX('
             148 (*modules | modules.ltxml)
             149 \newcommand\mhinputref[2][]{%
                   \def\@repos{#1}%
             150
                   \edef\mh@@repos{\mh@currentrepos}%
             151
                   \ifx\@repos\@empty%
             153
                   \else%
                     \mhcurrentrepos{#1}%
             154
             155
                   \inputref{\MathHub{\mh@currentrepos/source/#2}}%
             156
                   \mhcurrentrepos\mh@@repos%
             157
             158
                   \ignorespaces%
             159 }%
```

```
161 (modules.ltxml)');
   \mhinput
             162 (*modules)
             163 \let\mhinput\mhinputref%
             164 (/modules)
                     omtext-mh: MH Variants for OMText
              4.3
              We set up package options and pass them on to the omtext package, which we
              also load.
             165 (*omtext)
             166 \ProvidesPackage{omtext-mh}[2015/11/04 v1.0 MathHub support for the sTeX omtext package]
             167 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{omtext}}
             168 \ProcessOptions
             169 \RequirePackage{omtext}
             170 \RequirePackage{mathhub}
             171 (/omtext)
             172 (*omtext.ltxml)
             173 DeclareOption(undef,sub{PassOptions('omtext','sty',ToString(Digest(T_CS('\CurrentOption')))); }
             174 ProcessOptions();
             175 RequirePackage('omtext');
             176 RequirePackage('mathhub');
             177 (/omtext.ltxml)
\mh*graphics Use the current value of \mh@currentrepos or the value of the mhrepos key if it
              is given in \my*graphics.
             178 (*omtext)
             179 \addmetakey{Gin}{mhrepos}
             180 \newcommand\mhgraphics[2][]{\metasetkeys{Gin}{\#1}%
             181 \edef\mh@@repos{\mh@currentrepos}%
             182 \ ifx\Gin@mhrepos\empty\mygraphics[\#1]{\MathHub\{\mh@currentrepos/source/\#2\}\}\%}
             183 \else\mygraphics[#1]{\MathHub{\Gin@mhrepos/source/#2}}\fi
             184 \def\Gin@mhrepos{}\mhcurrentrepos\mh@@repos}
             185 \newcommand\mhcgraphics[2][]{\begin{center}\mhgraphics[#1]{#2}\end{center}}
             186 \newcommand\mhbgraphics[2][]{\fbox{\mhgraphics[#1]{#2}}}
             187 \newcommand\mhcbgraphics[2][]{\begin{center}\fbox{\mhgraphics[#1]{#2}}\end{center}}
             188 (/omtext)
             189 (*omtext.ltxml)
             190 sub mhgraphics {
                 my ($gullet,$keyval,$arg2) = @_;
                  my $repo_path;
                  if ($keyval) {
             193
                     $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
             194
                  if (! $repo_path) {
             195
                     $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }
             196
             197
                  else {
                     $keyval->setValue('mhrepos',undef); }
             198
```

160 (/modules | modules.ltxml)

```
my $mathhub_base = ToString(Digest('\MathHub{}'));
199
    my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);
    return Invocation(T_CS('\@includegraphicx'), $keyval, T_OTHER($finalpath)); }#$
202 DefKeyVal('Gin', 'mhrepos', 'Semiverbatim');
203 DefMacro('\mhgraphics OptionalKeyVals:Gin {}', \&mhgraphics);
204 DefMacro('\mhcgraphics []{}','\begin{center}\mhgraphics[#1]{#2}\end{center}');
205 DefMacro('\mhbgraphics []{}','\fbox{\mhgraphics[#1]{#2}}');
206 (/omtext.ltxml)
```

#### 4.4 smultiling-mh: MH Variants for Multilinguality

We set up package options and pass them on to the smultiling package, which we also load.

```
207 (*smultiling)
          208 \ProvidesPackage{smultiling-mh}[2015/11/04 v1.0 MathHub support for the sTeX smultiling package
          209 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{smultiling}}
          210 \ProcessOptions
          211 \RequirePackage{smultiling}
          212 \RequirePackage{mathhub}
          213 (/smultiling)
          214 (*smultiling.ltxml)
          215 DeclareOption(undef,sub{PassOptions('smultiling','sty',ToString(Digest(T_CS('\CurrentOption')))
          216 ProcessOptions();
          217 RequirePackage('smultiling');
          218 RequirePackage('mathhub');
          219 (/smultiling.ltxml)
mhmodnl:*
          220 (*smultiling)
          221 \addmetakey{mhmodnl}{repos}
          222 \addmetakey{mhmodnl}{path}
          223 \addmetakey*{mhmodnl}{title}
          224 \addmetakey*{mhmodnl}{creators}
          225 \addmetakey*{mhmodnl}{contributors}
          226 \addmetakey{mhmodnl}{srccite}
          227 \addmetakey{primary}{mhmodnl}[yes]
          228 (/smultiling)
          229 (*smultiling.ltxml)
          230 DefKeyVal('mhmodnl','title','Semiverbatim');
          231 DefKeyVal('mhmodnl', 'repos', 'Semiverbatim');
          232 DefKeyVal('mhmodnl','path','Semiverbatim');
          233 DefKeyVal('mhmodnl','creators','Semiverbatim');
          234 DefKeyVal('mhmodnl', 'contributors', 'Semiverbatim');
          235 DefKeyVal('mhmodnl', 'primary', 'Semiverbatim');
          236 (/smultiling.ltxml)
           The mhmodnl environment is just a layer over the module environment and the
  mhmodnl
```

\importmhmodule macro with the keys and language suitably adapted.

237 (\*smultiling)

```
239 \def\dest{#1}\ifx\dest\dempty\begin{module}[id=#2.#3]\else\begin{module}[id=#2.#3]\fif(module)]{ } id=#2.#3]\else\degin{module}[id=#2.#3]\fif(module)]{ } id=#2.#3]\fif(module)]{ 
                             241 \if@langfiles\importmhmodule[repos=\@repos,load=#2,ext=tex]{#2}\else
                             242 \ ifx\mhmodnl@load\\\end{empty} import\module {#2}\else\ import\module [ext=tex,load=\mhmodnl@load] {\#2}\fi%\mhmodnl@load\end{empty} import\module {$\#2$}\else\ import\module {$\#2$}\else\end{empty} import\module {$\#2$}\end{empty} import\module {$\#2$}\else\end{empty} import\module {$\#2$}\else\end{empty}
                             243 \fi}
                             244 {\end{module}}
                             245 (/smultiling)
                             246 (*smultiling.ltxml)
                             247 DefEnvironment('{mhmodnl} OptionalKeyVals:mhmodnl {}{}',
                                                                "?#excluded()(<omdoc:theory xml:id='#2.#3' >"
                                                                            "?&defined(&GetKeyVal(#1,'creators'))(<dc:creator>&GetKeyVal(#1,'creators')</dc:cr
                             249
                                                                            "?&defined(&GetKeyVal(#1,'title'))(<dc:title>&GetKeyVal(#1,'title')</dc:title>)()"
                             250
                                                                            "?&defined(&GetKeyVal(#1,'contributors'))(<dc:contributor>&GetKeyVal(#1,'contribut
                             251
                                                                            "<omdoc:imports from='?2&GetKeyVal(#1,'load')(&canonical_omdoc_path(&GetKeyVal(#1,'
                             252
                                                                            "#body"
                             253
                                                                 . "</omdoc:theory>)",
                             254
                                           afterDigestBegin=>sub {
                             255
                             256
                                                 my ($stomach, $whatsit) = @_;
                             257
                                                 my $keyval = $whatsit->getArg(1);
                             258
                                                 my $signature = ToString($whatsit->getArg(2));
                                                 my $language = ToString($whatsit->getArg(3));
                             259
                                                 my $repos = ToString(GetKeyVal($keyval,'torepos'));
                             260
                                                 my $current_repos = LookupValue('current_repos');
                             261
                                                 if (!$repos) { $repos = $current_repos; }
                             262
                                                 my $defpaths = LookupValue('defpath');
                             263
                                                 my $load_path = ($$defpaths{MathHub}).$repos.'/source/'.$signature;
                             264
                             265
                                                 if ($keyval) {
                             266
                                                       # If we're not given load, AND the langfiles option is in effect,
                             267
                                                       # default to #2
                             268
                             269
                                                       if ((! $keyval->getValue('path')) && (LookupValue('smultiling_langfiles'))) {
                             270
                                                             $keyval->setValue('load',$load_path); }
                             271
                                                       # Always load a TeX file
                                                       $keyval->setValue('ext','tex');
                             272
                                                       $keyval->setValue('id', "$signature.$language"); }
                             273
                                                 module_afterDigestBegin(@_);
                             274
                             275
                                                  importmoduleI(@_);
                                                 return; },
                             276
                                            afterDigest=>sub {
                             277
                             278
                                                 module_afterDigest(@_); });
                             279 (/smultiling.ltxml)%$
                              The mhviewsig environment is just a layer over the mhview environment with the
mhviewsig
                               keys suitably adapted.
                             280 \smultiling.ltxml\RawTeX('
                             281 (*smultiling | smultiling.ltxml)
                             282 \newenvironment{mhviewsig}[4][]{\def\@test{#1}\ifx\@test\@empty%
                             283 \begin{mhview}[id=#2,ext=tex]{#3}{#4}\else%
                             284 \begin{mhview}[id=#2,#1,ext=tex]{#3}{#4}\fi}
```

238 \newenvironment{mhmodnl}[3][]{\metasetkeys{mhmodnl}{#1}%

```
mhviewnl The mhviewnl environment is just a layer over the mhviewsketch environment
                    with the keys and language suitably adapted.<sup>4</sup>
                   286 \newenvironment{mhviewnl}[5][]{\def\@test{#1}\ifx\@test\@empty%
                   287 \begin{mhviewsketch}[id=#2.#5,ext=tex]{#3}{#4}\else%
                   288 \begin{mhviewsketch}[id=#2.#5,#1,ext=tex]{#3}{#4}\fi}
                   289 {\end{mhviewsketch}}
                   290 (/smultiling | smultiling.ltxml)
                   291 (smultiling.ltxml),;
                           structview-mh: MH Variants for Structures and Views
                    4.5
                    We set up package options and pass them on to the structview package, which
                    we also load.
                   292 (*structview)
                   293 \ProvidesPackage{structview-mh}[2015/11/04 v1.0 MathHub support for the sTeX structview package
                   294 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{structview}}
                   295 \ProcessOptions
                   296 \RequirePackage{structview}
                   297 \RequirePackage{mathhub}
                   298 (/structview)
                   299 (*structview.ltxml)
                   300 DeclareOption(undef, sub{PassOptions('structview', 'sty', ToString(Digest(T_CS('\CurrentOption')))
                   301 ProcessOptions();
                   302 RequirePackage('structview');
                   303 RequirePackage('mathhub');
                   304 (/structview.ltxml)
importmhmodulevia
                   305 \(\structview.ltxml\)\(\RawTeX(')
                   306 (*structview | structview.ltxml)
                   307 \newenvironment{importmhmodulevia}[3][]{%
                        \gdef\@doit{\importmhmodule[#1]{#2}{#3}}%
                        \ifmod@show\par\noindent importing module #2 via \@@doit\fi
                   310 }{%
                        \aftergroup\@@doit\ifmod@show end import\fi%
                   311
                   312 }%
                   313 (/structview | structview.ltxml)
                   314 (structview.ltxml)');
                   315 (*structview)
                   316 \srefaddidkey{mhview}
                   317 \addmetakey{mhview}{display}
                   318 \addmetakey{mhview}{creators}
                   319 \addmetakey{mhview}{contributors}
                   320 \addmetakey{mhview}{srccite}
```

285 {\end{mhview}}

EdN:4

non-trivial, since we do not know the current path, to which we could append . \( \lang \rangle !

 $^4\mathrm{EdNote}$ : MK: we have to do something about the if@langfiles situation here. But this is

```
321 \addmetakey*{mhview}{title}
       322 \addmetakey{mhview}{fromrepos}
       323 \addmetakey{mhview}{torepos}
       324 \addmetakey{mhview}{frompath}
       325 \addmetakey{mhview}{topath}
       326 \addmetakey[sms]{mhview}{ext}
       327 (/structview)
       328 (*structview.ltxml)
       329 DefKeyVal('mhview','id','Semiverbatim');
       330 DefKeyVal('mhview', 'display', 'Semiverbatim');
       331 DefKeyVal('mhview','creators','Semiverbatim');
       332 DefKeyVal('mhview','contributors','Semiverbatim');
       333 DefKeyVal('mhview','srccite','Semiverbatim');
       334 DefKeyVal('mhview','title','Semiverbatim');
       335 DefKeyVal('mhview', 'fromrepos', 'Semiverbatim');
       336 DefKeyVal('mhview','torepos','Semiverbatim');
       337 DefKeyVal('mhview', 'frompath', 'Semiverbatim');
       338 DefKeyVal('mhview', 'topath', 'Semiverbatim');
       339 DefKeyVal('mhview', 'ext', 'Semiverbatim');
       340 (/structview.ltxml)
mhview the MathHub version
       341 (*structview)
       342 \newenvironment{mhview}[3][]{% keys, from, to
            \metasetkeys{mhview}{#1}%
       343
            \sref@target%
       344
            \begin{@mhview}{#2}{#3}%
       345
            346
       347 }{%
            \end{@mhview}%
       349
            \ignorespaces%
       350 }%
       351 \ifmod@show\surroundwithmdframed{mhview}\fi
       352 (/structview)
       353 (*structview.ltxml)
       354 DefMacroI(T_CS('\begin{mhview}'),'OptionalKeyVals:mhview {}{}', sub {
            my ($gullet, $keyvals, $from_arg, $to_arg) = @_;
            my $from = ToString(Digest($from_arg));
       356
            my $to = ToString(Digest($to_arg));
       357
            AssignValue(from_module => $from);
       358
            AssignValue(to_module => $to);
       359
            my $from_repos = ToString(GetKeyVal($keyvals,'fromrepos'));
       360
            my $to_repos = ToString(GetKeyVal($keyvals,'torepos'));
       361
            my $repos = LookupValue('current_repos');
       362
            my $from_path = ToString(GetKeyVal($keyvals, 'frompath'));
       363
            my $to_path = ToString(GetKeyVal($keyvals, 'topath'));
       364
            my $ext = ToString(GetKeyVal($keyvals,'ext')) if $keyvals;
       365
            $ext = 'sms' unless $ext;
       366
       367
            my $current_repos = LookupValue('current_repos');
            if (!$from_repos) { $from_repos = $current_repos; }
```

```
if (!$to_repos) { $to_repos = $current_repos; }
             369
                  return (
             370
                     Tokenize("\\importMHmoduleI[repos=$from_repos,path=$from_path,ext=$ext]{$from}")->unlist,
             371
                     Tokenize("\\importMHmoduleI[repos=$to_repos,path=$to_path,ext=$ext]{$to}")->unlist,
             372
                     Invocation(T_CS('\begin{viewenv}'), $keyvals, $from_arg, $to_arg) ->unlist
             373
             374
                  );
             375 });
             376 DefMacroI('\end{mhview}',undef,'\end{viewenv}');
             377 (/structview.ltxml)
     Omhview The Omhview does the actual bookkeeping at the module level.
             378 (*structview)
             379 \newenvironment{@mhview}[2]{%from, to
                  \importmhmodule[repos=\mhview@fromrepos,path=\mhview@frompath,ext=\mhview@ext]{#1}%
                  \importmhmodule[repos=\mhview@torepos,path=\mhview@topath,ext=\mhview@ext]{#2}%
             382 }{}%
             383 (/structview)
mhviewsketch The mhviewsketch environment behaves like mhview, but only has text contents.
             384 (*structview)
             385 \newenvironment{mhviewsketch}[3][]{%
             386
                   \metasetkeys{mhview}{#1}%
                   \sref@target%
             387
                   \begin{@mhview}{#2}{#3}%
                   \view@heading{#2}{#3}{\mhview@display}{\mhview@title}%
             389
             390 }{%
                   \end{@mhview}%
             391
                   \ignorespaces%
             392
             393 }%
             394 \ifmod@show\surroundwithmdframed{mhviewsketch}\fi
             395 (/structview)
             396 (*structview.ltxml)
             397 DefMacroI(T_CS('\begin{mhviewsketch}'),'OptionalKeyVals:mhview {}{}', sub {
                  my ($gullet, $keyvals, $from_arg, $to_arg) = @_;
                  my $from = ToString(Digest($from_arg));
             399
                  my $to = ToString(Digest($to_arg));
             400
                  my $from_repos = ToString(GetKeyVal($keyvals,'fromrepos'));
                  my $to_repos = ToString(GetKeyVal($keyvals,'torepos'));
             402
                  my $repos = LookupValue('current_repos');
             403
                  my $from_path = ToString(GetKeyVal($keyvals,'frompath'));
             404
                  my $to_path = ToString(GetKeyVal($keyvals,'topath'));
             405
                  my $ext = ToString(GetKeyVal($keyvals,'ext')) if $keyvals;
             406
                  $ext = 'sms' unless $ext;
             407
                  my $current_repos = LookupValue('current_repos');
             408
                  if (!$from_repos) { $from_repos = $current_repos; }
             409
                  if (!$to_repos) { $to_repos = $current_repos; }
             410
                  return (
             411
                     Tokenize("\\importMHmoduleI[repos=$from_repos,path=$from_path,ext=$ext]{$from}")->unlist,
             412
             413
                     Tokenize("\\importMHmoduleI[repos=$to_repos,path=$to_path,ext=$ext]{$to}")->unlist,
             414
                     Invocation(T_CS('\begin{viewsketchenv}'), $keyvals, $from_arg, $to_arg)->unlist
```

```
415 ); 416 }); 417 DefMacroI('\end{mhviewsketch}',undef,'\end{viewsketchenv}'); 418 \langle /\text{structview.ltxml} \rangle
```

#### 4.6 mikoslides-mh: Support for MiKo Slides

```
We set up package options and pass them on to the mikoslides package, which we also load.
```

```
419 \( *mikoslides \)
420 \( Provides Package \{ mikoslides -mh \} [2015/11/04 v1.0 MathHub support for the sTeX mikoslides package \)
421 \( Declare Option * \{ Pass Options To Package \{ Current Option \} \{ mikoslides \} \)
422 \( Process Options \)
423 \( Require Package \{ mikoslides \} \)
424 \( Require Package \{ mathhub \} \)
425 \( / mikoslides \)
426 \( *mikoslides .ltxml \)
427 \( Declare Option (undef, sub \{ Pass Options ('mikoslides', 'sty', To String (Digest (T_CS ('\Current Option'))) \)
428 \( Process Options () \);
429 \( Require Package ('mikoslides') \);
430 \( Require Package ('mathhub') \);
431 \( / mikoslides .ltxml \)
```

\mhframeimage Use the current value of \mh@currentrepos or the value of the mhrepos key if it

443

444 }%

\fi%

445 (/mikoslides.ltxml | mikoslides)

446 (mikoslides.ltxml)');

```
is given in \frameimage.
432 \(\text{mikoslides}\)\\addmetakey{\(\mathbb{Gin}\)\}\(\mathbb{mhrepos}\)
433 (mikoslides.ltxml)DefKeyVal('Gin', 'mhrepos', 'Semiverbatim');
434 (mikoslides.ltxml)RawTeX('
435 (*mikoslides.ltxml | mikoslides)
436 \newcommand\mhframeimage[2][]{%
     <text>
437
     \edef\mh@@repos{\mh@currentrepos}%
438
439
     \ifx\Gin@mhrepos\@empty%
        \frameimage[#1]{\MathHub{\mh@currentrepos/source/#2}}%
440
441
     \else%
        \frameimage[#1]{\MathHub{\Gin@mhrepos/source/#2}}%
442
```

## 4.7 **problem-mh:** Support for Problems

We set up package options and pass them on to the problem package, which we also load.

```
447\ \langle *problem \rangle  
 448\ ProvidesPackage\{problem-mh\}[2015/11/04 v1.0 MathHub support for the sTeX problem package]  
 <math display="inline">449\ DeclareOption*{PassOptionsToPackage{\CurrentOption}{problem}}
```

```
450 \ProcessOptions
                  451 \RequirePackage{problem}
                  452 \RequirePackage{mathhub}
                  453 (/problem)
                  454 (*problem.ltxml)
                  455 DeclareOption(undef,sub{PassOptions('problem','sty',ToString(Digest(T_CS('\CurrentOption'))));
                  456 ProcessOptions();
                  457 RequirePackage('problem');
                  458 RequirePackage('mathhub');
                  459 (/problem.ltxml)
\includemhproblem
                   The \includemhproblem saves the current value of \mh@currentrepos in a local
                   macro \mh@@repos, resets \mh@currentrepos to the new value if one is given in
                   the optional argument, and after importing resets \mh@currentrepos to the old
                   value in \mh@@repos.
                  460 (*problem)
                  461 \newcommand\includemhproblem[2][]{\metasetkeys{inclprob}{#1}%
                  462 \edef\mh@currentrepos}\%
                  463 \ifx\inclprob@mhrepos\@empty\else\mhcurrentrepos\inclprob@mhrepos\fi%
                  464 \input{\MathHub{\mh@currentrepos/source/#2}}%
                  465 \mhcurrentrepos\mh@@repos\clear@inclprob@keys}
                  466 (/problem)
                  467 (*problem.ltxml)
                  468\;\mathrm{sub} includemhproblem {
                  469
                       my ($gullet,$keyval,$arg2) = @_;
                  470
                       my $repo_path;
                       if ($keyval) {
                  471
                  472
                          $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
                       if (! $repo_path) {
                  473
                          $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }
                  474
                  475
                       else {
                          $keyval->setValue('mhrepos',undef); }
                  476
                  477 my $mathhub_base = ToString(Digest('\MathHub{}'));
                  478 my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);
                  479 return Invocation(T_CS('\includeproblem'), $keyval, T_OTHER($finalpath)); }#$
                  480 DefKeyVal('inclprob', 'mhrepos', 'Semiverbatim');
                  481 DefMacro('\includemhproblem OptionalKeyVals:inclprob {}', \&includemhproblem);
                  482 (/problem.ltxml)
```

#### 4.8 hwexam-mh: Support for Assignments

We set up package options and pass them on to the hwexam package, which we also load.

```
483 (*hwexam)
484 \ProvidesPackage{hwexam-mh}[2015/11/04 v1.0 MathHub support for the sTeX hwexam package]
485 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{hwexam}}
486 \ProcessOptions
487 \RequirePackage{hwexam}
488 \RequirePackage{mathhub}
```

```
489 (/hwexam)
                     490 (*hwexam.ltxml)
                     491 DeclareOption(undef,sub{PassOptions('hwexam','sty',ToString(Digest(T_CS('\CurrentOption')))); }
                     492 ProcessOptions();
                     493 RequirePackage('hwexam');
                     494 RequirePackage('mathhub');
                     495 (/hwexam.ltxml)
\includemhassignment
                      The \includemhassignment saves the current value of \mh@currentrepos in a
                      local macro \mh@@repos, resets \mh@currentrepos to the new value if one is given
                      in the optional argument, and after importing resets \mh@currentrepos to the old
                      value in \mh@@repos.
                     496 (*package)
                     497 \newcommand\includemhassignment[2][]{\metasetkeys{inclassig}{#1}%
                     498 \edef\mh@@repos{\mh@currentrepos}%
                     499 \ifx\inclassig@mhrepos\@empty\else\mhcurrentrepos\inclassig@mhrepos\fi%
                     500 \includeassignment[#1]{\MathHub{\mh@currentrepos/source/#2}}%
                     501 \mhcurrentrepos\mh@@repos\clear@inclassig@keys}
                     502 (/package)
                     503 (*ltxml)
                     504 sub includemhassignment {
                          my ($gullet,$keyval,$arg2) = @_;
                          my $repo_path;
                     506
                          if ($keyval) {
                     507
                     508
                             $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
                     509
                          if (! $repo_path) {
                             $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }
                     510
                     511
                          else {
                             $keyval->setValue('mhrepos',undef); }
                     512
                          my $mathhub_base = ToString(Digest('\MathHub{}'));
                     513
                     514 my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);
                     return Invocation(T_CS('\includeassignment'), $keyval, T_OTHER($finalpath)); }#$
                     516 DefKeyVal('inclprob', 'mhrepos', 'Semiverbatim');
                     517 DefMacro('\includemhassignment OptionalKeyVals:inclprob {}', \&includemhassignment);
                     518 (/ltxml)
  \inputmhassignment analogous
                     519 (*package)
                     520 \newcommand\inputmhassignment[2][]{\metasetkeys{inclassig}{#1}%}
                     521 \edef\mh@currentrepos}%
                     522 \ifx\inclassig@mhrepos\@empty\else\mhcurrentrepos\inclassig@mhrepos\fi%
                     523 \inputassignment[#1] {\MathHub{\mh@currentrepos/source/#2}}%
                     524 \mhcurrentrepos\mh@@repos\clear@inclassig@keys}
                     525 (/package)
                     526 (*ltxml)
                     527 sub inputmhassignment {
                     528 my ($gullet,$keyval,$arg2) = @_;
                     529
                          my $repo_path;
                     530 if ($keyval) {
```

```
$repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
531
     if (! $repo_path) {
532
        $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }
533
     else {
534
       $keyval->setValue('mhrepos',undef); }
535
     my $mathhub_base = ToString(Digest('\MathHub{}'));
536
     my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);
537
    return Invocation(T_CS('\inputassignment'), $keyval, T_OTHER($finalpath)); }#$
539\ \texttt{DefMacro('\input mhassignment\ Optional Key Vals: inclprob\ \{\}',\ \& input mhassignment)};
540 \langle /ltxml \rangle
```

#### 4.9 Finale

Finally, we need to terminate the file with a success mark for perl.

```
541 \langle *ltxml \rangle
542 1;
543 \langle /ltxml \rangle
```

### References

- [Hor+11] Fulya Horozal et al. "Combining Source, Content, Presentation, Narration, and Relational Representation". In: *Intelligent Computer Mathematics*. Ed. by James Davenport et al. LNAI 6824. Springer Verlag, 2011, pp. 212-227. ISBN: 978-3-642-22672-4. URL: http://kwarc.info/frabe/Research/HIJKR\_dimensions\_11.pdf.
- [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).