# MathHub Support for STEX\*

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

December 4, 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	Inti	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 2015/11/22)

4.6	mikoslides-mh: Support for MiKo Slides								15
4.7	problem-mh: Support for Problems								16
4.8	hwexam-mh: Support for Assignments .								17
4.9	${\sf tikzinput\text{-}mh: Support \ for \ Assignments} \ \ .$								18
4.10	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 IATEX package (all the code between <code>\\*package</code>) and <code>\\*(package)</code>) and the IATEXML bindings (between <code>\\*Itxml</code>) and <code>\\*(Itxml</code>). 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.ltxml | structview.ltxml | omtext.ltxml | smultiling.ltxml | mikoslides.ltxml | problem.ltxml | hwexam.lt
2 # -*- CPERL -*-
3 package LaTeXML::Package::Pool;
4 use strict;
5 use LaTeXML::Package;
6 use LaTeXML::Util::Pathname;
7 (/ltxml | modules.ltxml | structview.ltxml | omtext.ltxml | smultiling.ltxml | mikoslides.ltxml | problem.ltxml | hwexam.lt
8 (package)\ProvidesPackage{mathhub}[2015/11/22 v1.0 sTeX Support for MathHub.info]
   Then we need to set up the packages by requiring the metakeys pack-
age [Koh15] to be loaded (in the right version).
9 (*package)
10 \RequirePackage{keyval}
11 (/package)
12 (*ltxml)
13 RequirePackage('keyval');
14 (/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.

```
15 (*package)
16 \newcommand\mhcurrentrepos[1]{%
    \ensuremath{\texttt{def}\@\text{test}{\#1}}\%
17
    \ifx\@test\mh@currentrepos% if new dir = old dir
18
19
      \relax% no need to change
20
      \protected@write\@auxout{}{\string\@mhcurrentrepos{#1}}%
21
22
    \@mhcurrentrepos{#1}% define mh@currentrepos
23
24 }%
25 \newcommand\@mhcurrentrepos[1]{\edef\mh@currentrepos{#1}}%
26 (/package)
27 (*ltxml)
28 DefMacro('\mhcurrentrepos{}','\@mhcurrentrepos{#1}');
29 DefMacro('\@mhcurrentrepos{#1}\0@mhcurrentrepos{#1}');
30 DefConstructor('\@@mhcurrentrepos{}','',
    afterDigest => sub{ AssignValue('current_repos',ToString($_[1]->getArg(1)),'global'); } );
32 (/ltxml)#$
```

```
the \libinput macro inputs from the lib directory of the MathHub repository
or the meta-inf/lib repos of the group.
33 (*package)
34 \def\modules@@first#1/#2;{#1}
35 \newcommand\libinput[1]{\def\@libfile{\MathHub{\mh@currentrepos/lib/#1}}%
36 \IfFileExists{\@libfile}{\input\@libfile}%
37 {\edef\@@group{\expandafter\modules@@first\mh@currentrepos;}
38 \edef\@inffile{\MathHub{\@@group/meta-inf/lib/#1}}
39 \IfFileExists{\@inffile}{\input{\@inffile}}%
40 {\PackageError{modules}
    {Library file missing, cannot input #1\MessageBreak%
42
      Both \@libfile.tex\MessageBreak and \@inffile.tex\MessageBreak do not exist}%
    {Check whether the file name is correct}}}}
43
44 (/package)
45 (*ltxml)
46 DefMacro('\modules@@first#1/#2;','#1');
47 DefMacro('\libinput {}', sub{
48
      my ($gullet, $name) = @_;
      my $mathhub_base = ToString(Digest('\MathHub{}'));
49
      my $repos = LookupValue('current_repos');
50
51
      # file name to search for
52
      $name = ToString($name);
53
      #Relative paths for recursive search
      my $reponame = substr($repos, 0, index($repos, '/'));
54
      my $FIRSTLIB = $mathhub_base . $repos . '/lib';
55
      my $SECONDLIB = $mathhub_base . $reponame . '/meta-inf/lib';
56
      my $file = pathname_find($name, types => ['tex'], paths =>[$FIRSTLIB]);
57
      $file = pathname_find($name, types=>['tex'], paths=>[$SECONDLIB]) unless $file;
58
      # Singal error if the file cannot be found
      LaTeXML::Package::InputContent($file, noerror=>1); });
61 (/ltxml)
```

#### 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.

```
62 \(\partial \text{*modules}\)
63 \(\text{ProvidesPackage{modules-mh}[2015/11/22 v1.0 MathHub support for the sTeX modules package]}\)
64 \(\text{RequirePackage{mathhub}}\)
65 \(\frac{\modules}\)
66 \(\text{*modules.ltxml}\)
67 \(\text{RequirePackage('mathhub');}\)
68 \(\frac{\modules.ltxml}\)
```

\importmhmodule

The \importmhmodule[\(\lambda key=value list\)]{\module} 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. We do all the \ifx compar-

ison with an \expandafter, since the values may be passed on from other key bindings. Parameters will be passed to \importmodule.

```
69 (*modules)
70 \srefaddidkey{importmhmodule}%
71 \addmetakey{importmhmodule}{repos}% saves the repo's path. E.g: smglom/numberfield
72 \addmetakey{importmhmodule}{path}% saves the module name. E.g: naturalnumbers
73 \addmetakey[sms]{importmhmodule}{ext}% saves the extension: E.g: tex
74 \addmetakey[false]{importmhmodule}{conservative}[true]%
75 \newcommand\importmhmodule[2][]{%
     \metasetkeys{importmhmodule}{#1}%
77
     \ifx\importmhmodule@path\@empty% if module name is not set
78
       \importmodule[ext=\importmhmodule@ext,id=\importmhmodule@id]{#2}%
79
       \edef\mh@@repos{\mh@currentrepos}% remember so that we can reset it.
80
81
       \ifx\importmhmodule@repos\@empty% if in the same repos
82
         \relax% no need to change mh@currentrepos, i.e, current dirctory.
83
         \mhcurrentrepos{\importmhmodule@repos}% change it.
84
85
       \importmodule[load=\MathHub{\mh@currentrepos/source/\importmhmodule@path},%
86
87
       ext=\importmhmodule@ext,id=\importmhmodule@id]{#2}%
88
       \mhcurrentrepos{\mh@@repos}% after importing, reset to old value
89
     \ignorespaces%
90
91 }%
92 (/modules)
93 (*modules.ltxml)
94 DefKeyVal('importmhmodule', 'id', 'Semiverbatim');
95 DefKeyVal('importmhmodule', 'repos', 'Semiverbatim');
96 DefKeyVal('importmhmodule', 'path', 'Semiverbatim');
97 DefKeyVal('importmhmodule', 'ext', 'Semiverbatim');
98 DefKeyVal('importmhmodule', 'conservative', 'Semiverbatim');
99 DefConstructor('\infty', importmhmodule OptionalKeyVals:importmhmodule {}',
          "<omdoc:imports "
100
101
          . "from='?%GetKeyVal(#1,'load')(&canonical_omdoc_path(&GetKeyVal(#1,'load')))()###2'"
102
                   . "?&defined(&GetKeyVal(#1,'conservative'))(load='&GetKeyVal(#1,'conservative')'
      afterDigest => \&importMHmoduleI);
103
104
105 sub importMHmoduleI {
     my ($stomach, $whatsit) = @_;
106
107
     my $keyval = $whatsit->getArg(1);
     my $id = $whatsit->getArg(2);
108
     if ($keyval) {
       my $repos = ToString($keyval->getValue('repos'));
110
       my $path = ToString($keyval->getValue('path'));
111
       my $current_repos = LookupValue('current_repos');
112
       if (!$repos) { # Use the implicit current repository
113
114
         $repos = $current_repos; }
115
       my $defpaths = LookupValue('defpath');
```

```
my $load_path = ($$defpaths{MathHub}).$repos.'/source/'.$path;
             116
                     $keyval->setValue('load',$load_path);
             117
                     AssignValue('current_repos' => $repos, 'global');
             118
                     importmoduleI($stomach,$whatsit);
             119
                     AssignValue('current_repos' => $current_repos, 'global'); }
             120
             121
             122
                     importmoduleI($stomach,$whatsit); }
             123
                   return; }
             124
             125 DefConstructor('\importMHmoduleI OptionalKeyVals:importmhmodule {}', '',
                    afterDigest=> \&importMHmoduleI );#$
              127 (/modules.ltxml)
              and now the analogs
\usemhmodule
             128 (*modules)
             129 \newcommand\usemhmodule[2][]{%
                   \metasetkeys{importmhmodule}{#1}%
                   \ifx\importmhmodule@path\@empty%
             131
                     \usemodule[ext=\importmhmodule@ext,id=\importmhmodule@id]{#2}%
             132
             133
             134
                     \edef\mh@@repos{\mh@currentrepos}%
             135
                     \ifx\importmhmodule@repos\@empty%
                     \else%
             136
                       \mhcurrentrepos{\importmhmodule@repos}%
             137
             138
             139
                     \usemodule[load=\MathHub{\mh@currentrepos/source/\importmhmodule@path},ext=\importmhmodule@
              140
                     \mhcurrentrepos\mh@@repos%
             141
                   \fi%
             142
                   \ignorespaces%
             143 }%
             144 (/modules)
             145 (*modules.ltxml)
             146 DefConstructor('\usemhmodule OptionalKeyVals:importmhmodule {}',
                    "<omdoc:uses from='?&GetKeyVal(#1,'load')(&canonical_omdoc_path(&GetKeyVal(#1,'load')))()###
                    afterDigest => \&importMHmoduleI);
             149 (/modules.ltxml)
\mhinputref
             150 \(\rangle\text{modules.ltxml}\)\(\rangle\text{RawTeX(')}\)
             151 (*modules | modules.ltxml)
             152 \newcommand\mhinputref[2][]{%
                   \def\@repos{#1}%
             153
                   \edef\mh@@repos{\mh@currentrepos}%
             154
                   \ifx\@repos\@empty%
             155
                   \else%
             156
                     \mhcurrentrepos{#1}%
             157
             158
                   \inputref{\MathHub{\mh@currentrepos/source/#2}}%
```

```
\ignorespaces%
             161
             162 }%
             163 (/modules | modules.ltxml)
             164 (modules.ltxml)');
    \mhinput
             165 (*modules)
             166 \let\mhinput\mhinputref%
             167 (/modules)
              4.3
                     omtext-mh: MH Variants for OMText
              We set up package options and pass them on to the omtext package, which we
              also load.
             168 (*omtext)
             169 \ProvidesPackage{omtext-mh}[2015/11/22 v1.0 MathHub support for the sTeX omtext package]
             170 \RequirePackage{mathhub}
             171 (/omtext)
             172 (*omtext.ltxml)
             173 RequirePackage('mathhub');
             174 \langle /omtext.ltxml \rangle
\mh*graphics Use the current value of \mh@currentrepos or the value of the mhrepos key if it
              is given in \my*graphics.
             175 (*omtext)
             176 \def\Gin@mhrepos{}
             177 \define@key{Gin}{mhrepos}{\csxdef\Gin@mhrepos{#1}}
             178 \newcommand\mhgraphics[2][]{\setkeys{Gin}{#1}%
             179 \edef\mh@@repos{\mh@currentrepos}%
             180 \ifx\Gin@mhrepos\@empty\mygraphics[#1]{\MathHub{\mh@currentrepos/source/#2}}%
             181 \else\mygraphics[#1]{\MathHub{\Gin@mhrepos/source/#2}}\fi
             182 \ensuremath{\tt lef\Gin\Qmhrepos\{}\mbcurrentrepos\mb\Qcrepos\}
             183 \newcommand\mhcgraphics[2][]{\begin{center}\mhgraphics[#1] {#2}\end{center}}
             184 \newcommand\mhbgraphics[2][]{\fbox{\mhgraphics[#1]{#2}}}
             185 \newcommand\mhcbgraphics[2][]{\begin{center}\fbox{\mhgraphics[#1]{#2}}\end{center}}
             186 (/omtext)
             187 (*omtext.ltxml)
             188 sub mhgraphics {
                  my ($gullet,$keyval,$arg2) = @_;
             190
                  my $repo_path;
             191
                  if ($keyval) {
                     $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
             192
             193
                   if (! $repo_path) {
                     $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }
             194
                   else {
             195
                     $keyval->setValue('mhrepos',undef); }
             196
             197
                   my $mathhub_base = ToString(Digest('\MathHub{}'));
                  my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);
```

\mhcurrentrepos\mh@@repos%

```
199 return Invocation(T_CS('\@includegraphicx'), $keyval, T_OTHER($finalpath)); }#$
200 DefKeyVal('Gin', 'mhrepos', 'Semiverbatim');
201 DefMacro('\mhgraphics OptionalKeyVals:Gin {}', \&mhgraphics);
202 DefMacro('\mhcgraphics []{}', '\begin{center}\mhgraphics[#1]{#2}\end{center}');
203 DefMacro('\mhbgraphics []{}', '\fbox{\mhgraphics[#1]{#2}}');
204 \/omtext.ltxml\
```

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

```
We set up package options and pass them on to the \mathsf{smultiling} package, which we also load.
```

```
205 \*smultiling\

206 \ProvidesPackage{smultiling-mh}[2015/11/22 v1.0 MathHub support for the sTeX smultiling package

207 \RequirePackage{mathhub}

208 \( /smultiling \)

209 \( *smultiling.ltxml \)

210 RequirePackage('mathhub');

211 \( /smultiling.ltxml \)
```

#### mhmodnl:\*

```
212 (*smultiling)
213 \addmetakey{mhmodnl}{repos}
214 \addmetakey{mhmodnl}{path}
215 \addmetakey*{mhmodnl}{title}
216 \addmetakey*{mhmodnl}{creators}
217 \addmetakey*{mhmodnl}{contributors}
218 \addmetakey{mhmodnl}{srccite}
219 \verb| \addmetakey{primary}{mhmodnl}[yes]|
220 (/smultiling)
221 (*smultiling.ltxml)
222 DefKeyVal('mhmodnl','title','Semiverbatim');
223 DefKeyVal('mhmodnl', 'repos', 'Semiverbatim');
224 DefKeyVal('mhmodnl','path','Semiverbatim');
225 DefKeyVal('mhmodnl','creators','Semiverbatim');
226 DefKeyVal('mhmodnl','contributors','Semiverbatim');
227 DefKeyVal('mhmodnl', 'primary', 'Semiverbatim');
228 (/smultiling.ltxml)
```

mhmodnl The mhmodnl environment is just a layer over the module environment and the \importmhmodule macro with the keys and language suitably adapted.

```
229 \*smultiling\
230 \newenvironment{mhmodnl}[3][]{\metasetkeys{mhmodnl}{#1}\%
231 \def\0test{#1}\ifx\0test\0empty\begin{module}[id=#2.#3]\else\begin{module}[id=#2.#3,#1]\fi\%
232 \edef\0repos{\ifx\mhmodnl@repos\0empty\mh@currentrepos\else\mhmodnl@repos\
233 \if@langfiles\importmhmodule[repos=\0repos,load=#2,ext=tex]{#2}\else
```

234 \ifx\mhmodnl@load\@empty\importmodule{#2}\else\importmodule[ext=tex,load=\mhmodnl@load]{#2}\fi% 235 \fi}

236 {\end{module}}
237 \square /smultiling \square

```
238 (*smultiling.ltxml)
          239 DefEnvironment('{mhmodnl} OptionalKeyVals:mhmodnl {}{}',
                       "?#excluded()(<omdoc:theory xml:id='#2.#3' >"
          240
                            "?&defined(&GetKeyVal(#1,'creators'))(<dc:creator>&GetKeyVal(#1,'creators')</dc:cr
          241
                           "?&defined(&GetKeyVal(#1,'title'))(<dc:title>&GetKeyVal(#1,'title')</dc:title>)()"
          242
                           "?&defined(&GetKeyVal(#1,'contributors'))(<dc:contributor>&GetKeyVal(#1,'contribut
          243
          244
                            "<omdoc:imports from='?&GetKeyVal(#1,'load')(&canonical_omdoc_path(&GetKeyVal(#1,'
                            "#body"
          245
                         "</omdoc:theory>)",
          246
                afterDigestBegin=>sub {
          247
                  my ($stomach, $whatsit) = @_;
          248
                  my $keyval = $whatsit->getArg(1);
          249
                  my $signature = ToString($whatsit->getArg(2));
          250
                  my $language = ToString($whatsit->getArg(3));
          251
                  my $repos = ToString(GetKeyVal($keyval,'torepos'));
          252
                  my $current_repos = LookupValue('current_repos');
          253
                  if (!$repos) { $repos = $current_repos; }
          254
                  my $defpaths = LookupValue('defpath');
          255
          256
                  my $load_path = ($$defpaths{MathHub}).$repos.'/source/'.$signature;
          257
          258
                  if ($keyval) {
                    # If we're not given load, AND the langfiles option is in effect,
          259
                    # default to #2
          260
                    if ((! $keyval->getValue('path')) && (LookupValue('smultiling_langfiles'))) {
          261
          ^{262}
                      $keyval->setValue('load',$load_path); }
                    # Always load a TeX file
          263
                    $keyval->setValue('ext','tex');
          264
                    $keyval->setValue('id', "$signature.$language"); }
          265
                  module_afterDigestBegin(@_);
          266
                  importmoduleI(@_);
          267
          268
                  return; },
          269
                afterDigest=>sub {
                  module_afterDigest(@_); });
          271 (/smultiling.ltxml)%$
           The mhviewsig environment is just a layer over the mhview environment with the
mhviewsig
           keys suitably adapted.
          272 \smultiling.ltxml\RawTeX('
          273 (*smultiling | smultiling.ltxml)
          274 \newenvironment{mhviewsig}[4][]{\def\@test{#1}\ifx\@test\@empty%
          275 \begin{mhview}[id=#2,ext=tex]{#3}{#4}\else%
          276 \begin{mhview}[id=#2,#1,ext=tex]{#3}{#4}\fi}
          277 {\end{mhview}}
           The mhviewnl environment is just a layer over the mhviewsketch environment
 mhviewnl
           with the keys and language suitably adapted.<sup>4</sup>
          278 \newenvironment{mhviewn1}[5][]{\def\0test{#1}\ifx\0test\0empty%
```

 $^4\mathrm{EdNote}$ : MK: we have to do something about the if@langfiles situation here. non-trivial, since we do not know the current path, to which we could append  $.\langle lang \rangle !$ 

EdN:4

```
279 \begin{mhviewsketch}[id=#2.#3,ext=tex]{#4}{#5}\else%
280 \begin{mhviewsketch}[id=#2.#3,#1,ext=tex]{#4}{#5}\fi}
281 {\end{mhviewsketch}}
282 (/smultiling | smultiling.ltxml)
283 \langle \text{smultiling.ltxml} \rangle,;
```

#### 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.
284 (*structview)
285 \ProvidesPackage{structview-mh}[2015/11/22 v1.0 MathHub support for the sTeX structview package
286 \RequirePackage{mathhub}
287 (/structview)
288 (*structview.ltxml)
289 RequirePackage('mathhub');
290 (/structview.ltxml)
291 \(\structview.\text{ltxml}\)\RawTeX(')
292 (*structview | structview.ltxml)
293 \newenvironment{importmhmodulevia}[3][]{%
     \gdef\@@doit{\importmhmodule[#1]{#2}{#3}}%
     \ifmod@show\par\noindent importing module #2 via \@@doit\fi
295
296 }{%
     \aftergroup\@@doit\ifmod@show end import\fi%
297
298 }%
299 (/structview | structview.ltxml)
300 (structview.ltxml)');
301 (*structview)
```

302 \srefaddidkey{mhview}

306 \addmetakey{mhview}{srccite}

307 \addmetakey\*{mhview}{title} 308 \addmetakey{mhview}{fromrepos}

309 \addmetakey{mhview}{torepos}  $310 \addmetakey{mhview}{frompath}$ 

311 \addmetakey{mhview}{topath}

312 \addmetakey[sms]{mhview}{ext}

313 (/structview)

importmhmodulevia

314 (\*structview.ltxml)

315 DefKeyVal('mhview','id','Semiverbatim'); 316 DefKeyVal('mhview', 'display', 'Semiverbatim');

317 DefKeyVal('mhview','creators','Semiverbatim'); 318 DefKeyVal('mhview','contributors','Semiverbatim');

319 DefKeyVal('mhview', 'srccite', 'Semiverbatim');

```
320 DefKeyVal('mhview', 'title', 'Semiverbatim');
        321 DefKeyVal('mhview', 'fromrepos', 'Semiverbatim');
        322 DefKeyVal('mhview','torepos','Semiverbatim');
        323 DefKeyVal('mhview','frompath','Semiverbatim');
        324 DefKeyVal('mhview', 'topath', 'Semiverbatim');
        325 DefKeyVal('mhview', 'ext', 'Semiverbatim');
        326 (/structview.ltxml)
mhview the MathHub version
        327 (*structview)
        328 \newenvironment{mhview}[3][]{% keys, from, to
             \metasetkeys{mhview}{#1}%
             \sref@target%
        330
             \begin{@mhview}{#2}{#3}%
        331
             332
        333 }{%
             \end{@mhview}%
        334
             \ignorespaces%
        335
        337 \ifmod@show\surroundwithmdframed{mhview}\fi
        338 (/structview)
        339 (*structview.ltxml)
        340 DefMacroI(T_CS('\begin{mhview}'), 'OptionalKeyVals:mhview {}{}', sub {
             my ($gullet, $keyvals, $from_arg, $to_arg) = @_;
             my $from = ToString(Digest($from_arg));
             my $to = ToString(Digest($to_arg));
        343
             AssignValue(from_module => $from);
        344
             AssignValue(to_module => $to);
        345
             my $from_repos = ToString(GetKeyVal($keyvals,'fromrepos'));
        346
             my $to_repos = ToString(GetKeyVal($keyvals,'torepos'));
        347
             my $repos = LookupValue('current_repos');
             my $from_path = ToString(GetKeyVal($keyvals,'frompath'));
        349
             my $to_path = ToString(GetKeyVal($keyvals, 'topath'));
        350
        351
             my $ext = ToString(GetKeyVal($keyvals,'ext')) if $keyvals;
             $ext = 'sms' unless $ext;
        352
        353
             my $current_repos = LookupValue('current_repos');
             if (!$from_repos) { $from_repos = $current_repos; }
        354
             if (!$to_repos) { $to_repos = $current_repos; }
        355
             return (
        356
               Tokenize("\\importMHmoduleI[repos=$from_repos,path=$from_path,ext=$ext]{$from}")->unlist,
        357
               Tokenize("\\importMHmoduleI[repos=$to_repos,path=$to_path,ext=$ext]{$to}")->unlist,
        358
               Invocation(T_CS('\begin{viewenv}'), $keyvals, $from_arg, $to_arg) -> unlist
        359
        360
            );
        361 });
        362 DefMacroI('\end{mhview}',undef,'\end{viewenv}');
        363 (/structview.ltxml)
Omhview The Omhview does the actual bookkeeping at the module level.
        364 (*structview)
        365 \newenvironment{@mhview}[2]{%from, to
```

```
\importmhmodule[repos=\mhview@torepos,path=\mhview@topath,ext=\mhview@ext]{#2}%
             367
             368 }{}%
             369 (/structview)
mhviewsketch The mhviewsketch environment behaves like mhview, but only has text contents.
             370 (*structview)
             371 \newenvironment{mhviewsketch}[3][]{%
                   \metasetkeys{mhview}{#1}%
                   \sref@target%
             373
                   \begin{@mhview}{#2}{#3}%
             374
                   \view@heading{#2}{#3}{\mhview@display}{\mhview@title}%
             375
             376 }{%
                   \end{@mhview}%
             377
             378
                   \ignorespaces%
             379 }%
             380 \ifmod@show\surroundwithmdframed{mhviewsketch}\fi
             381 (/structview)
             382 (*structview.ltxml)
             383 DefMacroI(T_CS('\begin{mhviewsketch}'), 'OptionalKeyVals:mhview {}{}', sub {
                   my ($gullet, $keyvals, $from_arg, $to_arg) = @_;
                   my $from = ToString(Digest($from_arg));
                  my $to = ToString(Digest($to_arg));
             386
                  my $from_repos = ToString(GetKeyVal($keyvals,'fromrepos'));
             387
                  my $to_repos = ToString(GetKeyVal($keyvals,'torepos'));
             388
                  my $repos = LookupValue('current_repos');
             389
                  my $from_path = ToString(GetKeyVal($keyvals,'frompath'));
                  my $to_path = ToString(GetKeyVal($keyvals,'topath'));
                  my $ext = ToString(GetKeyVal($keyvals,'ext')) if $keyvals;
             392
                   $ext = 'sms' unless $ext;
             393
                  my $current_repos = LookupValue('current_repos');
             394
                   if (!$from_repos) { $from_repos = $current_repos; }
             395
                   if (!$to_repos) { $to_repos = $current_repos; }
             396
                   return (
             397
                     Tokenize("\\importMHmoduleI[repos=$from_repos,path=$from_path,ext=$ext]{$from}")->unlist,
             398
             399
                     Tokenize("\\importMHmoduleI[repos=$to_repos,path=$to_path,ext=$ext]{$to}")->unlist,
                     Invocation(T_CS('\begin{viewsketchenv}'), $keyvals, $from_arg, $to_arg)->unlist
             400
             401
                  );
             402 });
             403 DefMacroI('\end{mhviewsketch}',undef,'\end{viewsketchenv}');
             404 (/structview.ltxml)
```

\importmhmodule[repos=\mhview@fromrepos,path=\mhview@frompath,ext=\mhview@ext]{#1}%

### 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.

```
405 (*mikoslides)
```

406 \ProvidesPackage{mikoslides-mh}[2015/11/22 v1.0 MathHub support for the sTeX mikoslides package 407 \RequirePackage{mathhub}

```
408 (/mikoslides)
                   409 (*mikoslides.ltxml)
                   410 RequirePackage('mathhub');
                   411 (/mikoslides.ltxml)
    \mhframeimage Use the current value of \mh@currentrepos or the value of the mhrepos key if it
                    is given in \frameimage.
                   412 (*mikoslides)
                   413 \def\Gin@mhrepos{}
                   414 \define@key{Gin}{mhrepos}{\csxdef\Gin@mhrepos{#1}}
                   415 (/mikoslides)
                   416 \(\text{mikoslides.ltxml}\)\)\)\)\DefKeyVal('Gin', 'mhrepos', 'Semiverbatim');
                   417 (mikoslides.ltxml)RawTeX('
                   418 \langle *mikoslides.ltxml \mid mikoslides \rangle
                   419 \newcommand\mhframeimage[2][]{%
                         \setkeys{Gin}{#1}%
                   420
                         \edef\mh@currentrepos}%
                   421
                         \ifx\Gin@mhrepos\@empty%
                   422
                           \frameimage[#1]{\MathHub{\mh@currentrepos/source/#2}}%
                   423
                   424
                         \else%
                           \frameimage[#1]{\MathHub{\Gin@mhrepos/source/#2}}%
                   425
                         \fi%
                   426
                   427 }%
                   428 (/mikoslides.ltxml | mikoslides)
                   429 (mikoslides.ltxml),);
                    4.7
                           problem-mh: Support for Problems
                    We set up package options and pass them on to the problem package, which we
                    also load.
                   430 (*problem)
                   431 \ProvidesPackage{problem-mh}[2015/11/22 v1.0 MathHub support for the sTeX problem package]
                   432 \RequirePackage{mathhub}
                   433 (/problem)
                   434 (*problem.ltxml)
                   435 RequirePackage('mathhub');
                   436 (/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
```

 $440 \verb|\inclprob@mhrepos\\empty\\else\\mhcurrentrepos\\inclprob@mhrepos\\fi%$ 

438 \newcommand\includemhproblem[2][]{\metasetkeys{inclprob}{#1}%

value in \mh@@repos.

439 \edef\mh@@repos{\mh@currentrepos}%

441 \input{\MathHub{\mh@currentrepos/source/#2}}% 442 \mhcurrentrepos\mh@@repos\clear@inclprob@keys}

437 (\*problem)

443 (/problem)

```
444 (*problem.ltxml)
445 \; \mathrm{sub} \; \; \mathrm{includemhproblem} \; \{
    my ($gullet,$keyval,$arg2) = @_;
    my $repo_path;
447
448
    if ($keyval) {
       $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
449
450
     if (! $repo_path) {
       $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }
451
    else {
452
       $keyval->setValue('mhrepos',undef); }
453
    my $mathhub_base = ToString(Digest('\MathHub{}'));
454
     my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);
     return Invocation(T_CS('\includeproblem'), $keyval, T_OTHER($finalpath)); }#$
457 DefKeyVal('inclprob', 'mhrepos', 'Semiverbatim');
458 DefMacro('\includemhproblem OptionalKeyVals:inclprob {}', \&includemhproblem);
459 (/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.

\includemhassignment

The \includemhassignment saves the current value of \mh@currentrepos in a local macro \mh@currentrepos, 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@currentrepos.

```
467 (*hwexam)
468 \newcommand\includemhassignment[2][]{\metasetkeys{inclassig}{#1}\%
469 \edef\mh@currentrepos}%
470 \ifx\inclassig@mhrepos\@empty\else\mhcurrentrepos\inclassig@mhrepos\fi%
471 \includeassignment [#1] {\MathHub{\mh@currentrepos/source/#2}}%
472 \mhcurrentrepos\mh@@repos\clear@inclassig@keys}
473 (/hwexam)
474 (*hwexam.ltxml)
475 sub includemhassignment {
    my ($gullet,$keyval,$arg2) = @_;
    my $repo_path;
    if ($keyval) {
478
479
       $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
480
    if (! $repo_path) {
       $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }
481
482
    else {
```

```
$keyval->setValue('mhrepos',undef); }
                   483
                        my $mathhub_base = ToString(Digest('\MathHub{}'));
                   484
                        my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);
                   486 return Invocation(T_CS('\includeassignment'), $keyval, T_OTHER($finalpath)); }#$
                   487 DefKeyVal('inclprob', 'mhrepos', 'Semiverbatim');
                   488 DefMacro('\includemhassignment OptionalKeyVals:inclprob {}', \&includemhassignment);
                   489 (/hwexam.ltxml)
\inputmhassignment analogous
                   490 (*hwexam)
                   491 \newcommand\inputmhassignment[2][]{\metasetkeys{inclassig}{#1}%
                   492 \edef\mh@currentrepos}%
                   493 \ifx\inclassig@mhrepos\@empty\else\mhcurrentrepos\inclassig@mhrepos\fi%
                   494 \inputassignment[#1]{\MathHub{\mh@currentrepos/source/#2}}%
                   495 \mhcurrentrepos\mh@@repos\clear@inclassig@keys}
                   496 (/hwexam)
                   497 (*hwexam.ltxml)
                   498 sub inputmhassignment {
                        my ($gullet,$keyval,$arg2) = 0_;
                        my $repo_path;
                   500
                        if ($keyval) {
                   501
                          $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
                   502
                        if (! $repo_path) {
                   504
                          $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }
                   505
                        else {
                          $keyval->setValue('mhrepos',undef); }
                   506
                        my $mathhub_base = ToString(Digest('\MathHub{}'));
                   507
                        my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);
                       return Invocation(T_CS('\inputassignment'), $keyval, T_OTHER($finalpath)); }#$
                   510 DefMacro('\inputmhassignment OptionalKeyVals:inclprob {}', \&inputmhassignment);
                   511 (/hwexam.ltxml)
```

### 4.9 tikzinput-mh: Support for Assignments

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

```
512 \*tikzinput\\
513 \ProvidesPackage{tikzinput-mh}[2015/11/22 v1.0 MathHub support for the sTeX tikzinput package]
514 \RequirePackage{mathhub}
515 \(/tikzinput\)
516 \(*tikzinput.ltxml\)
517 \RequirePackage('mathhub');
518 \(/tikzinput.ltxml\)
519 \(\tikzinput.ltxml\)\RawTeX('
520 \(*tikzinput.ltxml\)\RawTeX('
521 \\define@key{Gin}{mhrepos}{\csxdef\Gin@mhrepos{#1}}\)
522 \newcommand\mhtikzinput[2][]{\def\Gin@mhrepos{}\setkeys{Gin}{#1}\\\
523 \\def\mh@crepos{\mh@currentrepos}\\\\
524 \\ifx\Gin@mhrepos\@empty\tikzinput[#1]{\MathHub{\mh@currentrepos/source/#2}}\\\\\\
```

```
525 \else\tikzinput[#1]{\MathHub{\Gin@mhrepos/source/#2}}\fi
526 \def\Gin@mhrepos{}\mhcurrentrepos\mh@repos}
527 \newcommand\cmhtikzinput[2][]{\begin{center}\mhtikzinput[#1]{#2}\end{center}}
528 \langle \tikzinput | tikzinput.ltxml\rangle
529 \langle \tikzinput.ltxml\rangle
);
```

### 4.10 Finale

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

 $530 \ \langle ltxml \ | \ modules.ltxml \ | \ structview.ltxml \ | \ omtext.ltxml \ | \ smultiling.ltxml \ | \ mikoslides.ltxml \ | \ problem.ltxml \ | \ hwexam.ltxml \$ 

# 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).