

MathHub Support for \LaTeX^*

Michael Kohlhasse
Jacobs University, Bremen
<http://kwarc.info/kohlhasse>

November 22, 2015

Abstract

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

The `mathhub` packages extend \LaTeX with support for the MathHub.info portal

Contents

1	Introduction	3
2	The User Interface	3
2.1	Package Options	3
2.2	<code>modules-mh</code> : MH Variants for Modules	3
2.3	<code>omtext-mh</code> : MH Variants for OMText	4
2.4	<code>statements-mh</code> : MH Variants for Statements	4
2.5	<code>smultiling-mh</code> : MH Variants for Multilinguality	4
2.6	<code>structview-mh</code> : MH Variants for Structures and Views	4
2.7	<code>mikoslides-mh</code> : Support for MiKo Slides	5
2.8	<code>problem-mh</code> : Support for Problems	5
2.9	<code>hwexam-mh</code> : Support for Assignments	5
3	Limitations	6
4	Implementation	7
4.1	General Infrastructure	7
4.2	<code>modules-mh</code> : MH Variants for Modules	8
4.3	<code>omtext-mh</code> : MH Variants for OMText	11
4.4	<code>statements-mh</code> : MH Variants for Statements	12

*Version v1.0 (last revised 2015/11/22)

4.5	<code>smultiling-mh</code> : MH Variants for Multilinguality	12
4.6	<code>structview-mh</code> : MH Variants for Structures and Views	14
4.7	<code>mikoslides-mh</code> : Support for MiKo Slides	17
4.8	<code>problem-mh</code> : Support for Problems	17
4.9	<code>hwexam-mh</code> : Support for Assignments	18
4.10	<code>tikzinput-mh</code> : Support for Assignments	20
4.11	Finale	20

1 Introduction

Much of the \LaTeX 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 \LaTeX author with **MathHub**-enabled versions of the \LaTeX 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 be loaded, `\importmhmodule` is the same as `\importmodule`.

`\mhcurrentrepos` Of course, neither \LaTeX nor \LaTeXML know 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 each module, since the `\importmhmodule` macro sets the current repository automatically.

`\usemhmodule` The `\usemhmodule` is the analog to `\usemodule`.

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

2.3 omtex-mh: MH Variants for OMText

`\mhgraphics` The `\mhgraphics` 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)

```
\mhgraphics[fooMH/bar]{baz/foobar}
```

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

2.4 statements-mh: MH Variants for Statements

this only provides `\usemhvocab` a variant of `\usevocab` (which might go away at some time)

2.5 smultiling-mh: MH Variants for Multilinguality

1 2

2.6 structview-mh: MH Variants for Structures and Views

3

EdN:1
EdN:2

EdN:3

2.7 mikoslides-mh: Support for MiKo Slides

`\mhframeimage` 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}}
```

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}
```

2.8 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`.

2.9 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}
```

¹EDNOTE: needs to be documented

²EDNOTE: mhmodsig seems to be missing what happened?

³EDNOTE: needs to be documented

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 `sTeX` GitHub repository [sTeX].

1. none reported yet.

4 Implementation

The `sref` package generates two files: the \LaTeX package (all the code between `<*package>` and `</package>`) and the \LaTeX XML bindings (between `<*ltxml>` and `</ltxml>`). 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 \LaTeX XML binding files in the base package.

```
1 <*ltxml | modules.ltxml | structview.ltxml | omtex.ltxml | statements.ltxml | smultiling.ltxml | mikosides.ltxml | problem
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 | omtex.ltxml | statements.ltxml | smultiling.ltxml | mikosides.ltxml | problem
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` package [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` 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]{%
17   \edef\@test{#1}%
18   \ifx\@test\mh@currentrepos% if new dir = old dir
19     \relax% no need to change
20   \else%
21     \protected@write\@auxout{}{\string\@mhcurrentrepos{#1}}%
22   \fi%
23   \@mhcurrentrepos{#1}% define mh@currentrepos
24 }%
25 \newcommand\@mhcurrentrepos[1]{\edef\mh@currentrepos{#1}}%
26 </package>
27 <*ltxml>
28 \DefMacro(' \mhcurrentrepos{ }', '\@mhcurrentrepos{#1}');
29 \DefMacro(' \@mhcurrentrepos{ }', '\def\mh@currentrepos{#1}\@mhcurrentrepos{#1}');
30 \DefConstructor(' \@mhcurrentrepos{ }', '',
31   afterDigest => sub{ AssignValue('current_repos', ToString($_[1]->getArg(1)), 'global'); } );
32 </ltxml>#</pre>
```

```

\libinput 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}
41 {Library file missing, cannot input #1\MessageBreak%
42 Both \@libfile.tex\MessageBreak and \@inffile.tex\MessageBreak do not exist}%
43 {Check whether the file name is correct}}}%
44 </package>
45 <*txml>
46 DefMacro('\modules@@first#1/#2;', '#1');
47 DefMacro('\libinput {}', sub{
48 my ($gullet, $name) = @_ ;
49 my $mathhub_base = ToString(Digest('\MathHub{'}));
50 my $repos = LookupValue('current_repos');
51 # file name to search for
52 $name = ToString($name);
53 #Relative paths for recursive search
54 my $reponame = substr($repos, 0, index($repos, '/'));
55 my $FIRSTLIB = $mathhub_base . $repos . '/lib' ;
56 my $SECONDLIB = $mathhub_base . $reponame . '/meta-inf/lib';
57 my $file = pathname_find($name, types => ['tex'], paths => [$FIRSTLIB]);
58 $file = pathname_find($name, types=>['tex'], paths=>[$SECONDLIB]) unless $file;
59 # Singal error if the file cannot be found
60 LaTeXML::Package::InputContent($file, noerror=>1); });
61 </txml>

```

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

```

`\importmhmodule` The `\importmhmodule[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][]{%
76   \metasetkeys{importmhmodule}{#1}%
77   \ifx\importmhmodule@path@empty% if module name is not set
78     \importmodule[ext=\importmhmodule@ext,id=\importmhmodule@id]{#2}%
79   \else%
80     \edef\mh@crepos{\mh@currentrepos}% remember so that we can reset it.
81     \ifx\importmhmodule@repos@empty% if in the same repos
82       \relax% no need to change mh@currentrepos, i.e, current directory.
83     \else%
84       \mhcurrentrepos{\importmhmodule@repos}% change it.
85     \fi%
86     \importmodule[load=\MathHub{\mh@currentrepos/source/\importmhmodule@path},%
87       ext=\importmhmodule@ext,id=\importmhmodule@id]{#2}%
88     \mhcurrentrepos{\mh@crepos}% after importing, reset to old value
89   \fi%
90   \ignorespaces%
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('\importmhmodule OptionalKeyVals:importmhmodule {}',
100   "<omdoc:imports "
101   . "from='?&GetKeyVal(#1,'load')(&canonical_omdoc_path(&GetKeyVal(#1,'load')))(###2'"
102   . "?&defined(&GetKeyVal(#1,'conservative'))(load='&GetKeyVal(#1,'conservative'))'"
103   afterDigest => \&importMHmoduleI);
104
105 sub importMHmoduleI {
106   my ($stomach, $whatsit) = @_;
107   my $keyval = $whatsit->getArg(1);
108   my $id = $whatsit->getArg(2);
109   if ($keyval) {
110     my $repos = ToString($keyval->getValue('repos'));
111     my $path = ToString($keyval->getValue('path'));
112     my $current_repos = LookupValue('current_repos');
113     if (!$repos) { # Use the implicit current repository
114       $repos = $current_repos; }
115     my $defpaths = LookupValue('defpath');

```

```

116 my $load_path = ($$defpaths{MathHub}).$repos.'/source/'$.path;
117 $keyval->setValue('load',$load_path);
118 AssignValue('current_repos' => $repos, 'global');
119 importmoduleI($stomach,$whatsit);
120 AssignValue('current_repos' => $current_repos, 'global'); }
121 else {
122   importmoduleI($stomach,$whatsit); }
123 return; }
124
125 DefConstructor('\importMHmoduleI OptionalKeyVals:importmhmodule {}', '',
126   afterDigest=> \&importMHmoduleI );#$
127 </modules.ltxml>

```

and now the analogs

\usemhmodule

```

128 <*modules>
129 \newcommand\usemhmodule[2] [] {%
130   \metasetkeys{importmhmodule}{#1}%
131   \ifx\importmhmodule@path\empty%
132     \usemodule[ext=\importmhmodule@ext,id=\importmhmodule@id]{#2}%
133   \else%
134     \edef\mh@@repos{\mh@currentrepos}%
135     \ifx\importmhmodule@repos\empty%
136       \else%
137         \mhcurrentrepos{\importmhmodule@repos}%
138       \fi%
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 {}',
147   "<omdoc:uses from='?&GetKeyVal(#1,'load')(&canonical_omdoc_path(&GetKeyVal(#1,'load')))(###
148   afterDigest => \&importMHmoduleI);
149 </modules.ltxml>

```

\mhinputref

```

150 <modules.ltxml>RawTeX(
151 <*modules | modules.ltxml>
152 \newcommand\mhinputref[2] [] {%
153   \def\@repos{#1}%
154   \edef\mh@@repos{\mh@currentrepos}%
155   \ifx\@repos\empty%
156     \else%
157       \mhcurrentrepos{#1}%
158     \fi%
159   \inputref{\MathHub{\mh@currentrepos/source/#2}}%

```

```

160 \mhcurrentrepos\mh@crepos%
161 \ignorespaces%
162 }%
163 </modules | modules.ltxml>
164 <modules.ltxml>' );

```

\mhinput

```

165 <*modules>
166 \let\mhinput\mhinputref%
167 </modules>

```

4.3 omtex-mh: MH Variants for OMTex

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

```

168 <*omtext>
169 \ProvidesPackage{omtex-mh}[2015/11/22 v1.0 MathHub support for the sTeX omtex package]
170 \RequirePackage{mathhub}
171 </omtex>
172 <*omtex.ltxml>
173 \RequirePackage('mathhub');
174 </omtex.ltxml>

```

\mh*graphics Use the current value of \mh@currentrepos or the value of the mhrepos key if it is given in \my*graphics.

```

175 <*omtex>
176 \def\Gin@mhrepos{}
177 \define@key{Gin}{mhrepos}{\csxdef\Gin@mhrepos{#1}}
178 \newcommand\mhgraphics[2] [] {\setkeys{Gin}{#1}%
179 \edef\mh@crepos{\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 \def\Gin@mhrepos{}\mhcurrentrepos\mh@crepos}
183 \newcommand\mhgraphics[2] [] {\begin{center}\mhgraphics[#1]{#2}\end{center}}
184 \newcommand\mhgraphics[2] [] {\fbox{\mhgraphics[#1]{#2}}}
185 \newcommand\mhgraphics[2] [] {\begin{center}\fbox{\mhgraphics[#1]{#2}}\end{center}}
186 </omtex>
187 <*omtex.ltxml>
188 sub mhgraphics {
189   my ($gullet,$keyval,$arg2) = @_ ;
190   my $repo_path;
191   if ($keyval) {
192     $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
193   if (! $repo_path) {
194     $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }
195   else {
196     $keyval->setValue('mhrepos',undef); }
197   my $mathhub_base = ToString(Digest('\MathHub{'}));
198   my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);

```

```

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 statements-mh: MH Variants for Statements

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

```

205 <*statements>
206 \ProvidesPackage{statements-mh}[2015/11/22 v1.0 MathHub support for the sTeX statements package]
207 \RequirePackage{mathhub}
208 </statements>
209 <*statements.ltxml>
210 RequirePackage('mathhub');
211 </statements.ltxml>

212 <*statements>
213 \let\usemhvocab=\usemhmodule
214 </statements>
215 <*statements.ltxml>
216 DefMacro('\usemhvocab', '\usemhmodule');
217 </statements.ltxml>

```

4.5 smultiling-mh: MH Variants for Multilinguality

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

```

218 <*smultiling>
219 \ProvidesPackage{smultiling-mh}[2015/11/22 v1.0 MathHub support for the sTeX smultiling package]
220 \RequirePackage{mathhub}
221 </smultiling>
222 <*smultiling.ltxml>
223 RequirePackage('mathhub');
224 </smultiling.ltxml>

```

`mhmodnl:*`

```

225 <*smultiling>
226 \addmetakey{mhmodnl}{repos}
227 \addmetakey{mhmodnl}{path}
228 \addmetakey*{mhmodnl}{title}
229 \addmetakey*{mhmodnl}{creators}
230 \addmetakey*{mhmodnl}{contributors}
231 \addmetakey{mhmodnl}{srccite}
232 \addmetakey{primary}{mhmodnl}[yes]
233 </smultiling>
234 <*smultiling.ltxml>

```

```

235 DefKeyVal('mhmodnl','title','Semiverbatim');
236 DefKeyVal('mhmodnl','repos','Semiverbatim');
237 DefKeyVal('mhmodnl','path','Semiverbatim');
238 DefKeyVal('mhmodnl','creators','Semiverbatim');
239 DefKeyVal('mhmodnl','contributors','Semiverbatim');
240 DefKeyVal('mhmodnl','primary','Semiverbatim');
241 </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.

```

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

```

```

282 afterDigest=>sub {
283   module_afterDigest(@_); });
284 </smultiling.ltxml>%$

```

mhviewsig The **mhviewsig** environment is just a layer over the **mhview** environment with the keys suitably adapted.

```

285 <smultiling.ltxml>RawTeX('
286 <*smultiling | smultiling.ltxml>
287 \newenvironment{mhviewsig}[4][\def\@test{#1}\ifx\@test\@empty%
288 \begin{mhview}[id=#2,ext=tex]{#3}{#4}\else%
289 \begin{mhview}[id=#2,#1,ext=tex]{#3}{#4}\fi}
290 {\end{mhview}}

```

mhviewnl The **mhviewnl** environment is just a layer over the **mhviewsketch** environment with the keys and language suitably adapted.⁴

```

291 \newenvironment{mhviewnl}[5][\def\@test{#1}\ifx\@test\@empty%
292 \begin{mhviewsketch}[id=#2.#5,ext=tex]{#3}{#4}\else%
293 \begin{mhviewsketch}[id=#2.#5,#1,ext=tex]{#3}{#4}\fi}
294 {\end{mhviewsketch}}
295 </smultiling | smultiling.ltxml>
296 <smultiling.ltxml>');

```

4.6 structview-mh: MH Variants for Structures and Views

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

```

297 <*structview>
298 \ProvidesPackage{structview-mh}[2015/11/22 v1.0 MathHub support for the sTeX structview package]
299 \RequirePackage{mathhub}
300 </structview>
301 <*structview.ltxml>
302 \RequirePackage('mathhub');
303 </structview.ltxml>

```

importmhmodulevia

```

304 <structview.ltxml>RawTeX('
305 <*structview | structview.ltxml>
306 \newenvironment{importmhmodulevia}[3][\%
307   \gdef\@doit{\importmhmodule[#1]{#2}{#3}}%
308   \ifmod@show\par\noindent importing module #2 via \@doit\fi
309 ]{\%
310   \aftergroup\@doit\ifmod@show end import\fi%
311 }%
312 </structview | structview.ltxml>
313 <structview.ltxml>');

```

⁴EDNOTE: MK: we have to do something about the `if@langfiles` situation here. But this is non-trivial, since we do not know the current path, to which we could append `.\lang`!

```

314 <*structview>
315 \srefaddidkey{mhview}
316 \addmetakey{mhview}{display}
317 \addmetakey{mhview}{creators}
318 \addmetakey{mhview}{contributors}
319 \addmetakey{mhview}{srccite}
320 \addmetakey*{mhview}{title}
321 \addmetakey{mhview}{fromrepos}
322 \addmetakey{mhview}{torepos}
323 \addmetakey{mhview}{frompath}
324 \addmetakey{mhview}{topath}
325 \addmetakey[sms]{mhview}{ext}
326 </structview>
327 <*structview.ltxml>
328 DefKeyVal('mhview','id','Semiverbatim');
329 DefKeyVal('mhview','display','Semiverbatim');
330 DefKeyVal('mhview','creators','Semiverbatim');
331 DefKeyVal('mhview','contributors','Semiverbatim');
332 DefKeyVal('mhview','srccite','Semiverbatim');
333 DefKeyVal('mhview','title','Semiverbatim');
334 DefKeyVal('mhview','fromrepos','Semiverbatim');
335 DefKeyVal('mhview','torepos','Semiverbatim');
336 DefKeyVal('mhview','frompath','Semiverbatim');
337 DefKeyVal('mhview','topath','Semiverbatim');
338 DefKeyVal('mhview','ext','Semiverbatim');
339 </structview.ltxml>

```

mhview the MathHub version

```

340 <*structview>
341 \newenvironment{mhview}[3][{}]{% keys, from, to
342 \metasetkeys{mhview}{#1}%
343 \sref@target%
344 \begin{@mhview}{#2}{#3}%
345 \view@heading{#2}{#3}{\mhview@display}{\mhview@title}%
346 }{%
347 \end{@mhview}%
348 \ignorespaces%
349 }%
350 \ifmod@show\surroundwithmdframed{mhview}\fi
351 </structview>
352 <*structview.ltxml>
353 DefMacroI(T_CS('\begin{mhview}'), 'OptionalKeyVals: mhview {}{}', sub {
354 my ($gullet, $keyvals, $from_arg, $to_arg) = @_;
355 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 my $current_repos = LookupValue('current_repos');
367 if (!$from_repos) { $from_repos = $current_repos; }
368 if (!$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 DefMacroI('\end{mhview}',undef,'\end{viewenv}');
376 </structview.ltxml>

```

@mhview The @mhview does the actual bookkeeping at the module level.

```

377 <*structview>
378 \newenvironment{@mhview}[2]{%from, to
379   \importmhmodule[repos=\mhview@fromrepos,path=\mhview@frompath,ext=\mhview@ext]{#1}%
380   \importmhmodule[repos=\mhview@torepos,path=\mhview@topath,ext=\mhview@ext]{#2}%
381 }{}%
382 </structview>

```

mhviewsketch The mhviewsketch environment behaves like mhview, but only has text contents.

```

383 <*structview>
384 \newenvironment{mhviewsketch}[3][{%
385   \metasetkeys{mhview}{#1}%
386   \sref@target%
387   \begin{@mhview}{#2}{#3}%
388   \view@heading{#2}{#3}{\mhview@display}{\mhview@title}%
389 }{%
390   \end{@mhview}%
391   \ignorespaces%
392 }%
393 \ifmod@show\surroundwithmdframed{mhviewsketch}\fi
394 </structview>
395 <*structview.ltxml>
396 DefMacroI(T_CS('\begin{mhviewsketch}'),'OptionalKeyVals:mhview {}{}', sub {
397   my ($gullet, $keyvals, $from_arg, $to_arg) = @_;
398   my $from = ToString(Digest($from_arg));
399   my $to = ToString(Digest($to_arg));
400   my $from_repos = ToString(GetKeyVal($keyvals,'fromrepos'));
401   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]{$from}")->unlist,
412   Tokenize("\importMHmoduleI[repos=$to_repos,path=$to_path,ext=$ext]{$to}")->unlist,
413   Invocation(T_CS('\begin{viewsketchenv}'),$keyvals,$from_arg,$to_arg)->unlist
414 );
415 });
416 DefMacroI('\end{mhviewsketch}',undef,'\end{viewsketchenv}');
417 </structview.ltxml>

```

4.7 mikoslides-mh: Support for MiKo Slides

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

```

418 <*mikoslides>
419 \ProvidesPackage{mikoslides-mh}[2015/11/22 v1.0 MathHub support for the sTeX mikoslides package]
420 \RequirePackage{mathhub}
421 </mikoslides>
422 <*mikoslides.ltxml>
423 \RequirePackage{'mathhub'};
424 </mikoslides.ltxml>

```

`\mhframeimage` Use the current value of `\mh@currentrepos` or the value of the `mhrepos` key if it is given in `\frameimage`.

```

425 <*mikoslides>
426 \def\Gin@mhrepos{}
427 \define@key{Gin}{mhrepos}{\csxdef\Gin@mhrepos{#1}}
428 </mikoslides>
429 <mikoslides.ltxml>\DefKeyVal{'Gin','mhrepos','Semiverbatim'};
430 <mikoslides.ltxml>\RawTeX('
431 <*mikoslides.ltxml | mikoslides>
432 \newcommand\mhframeimage[2][{}]{%
433   \setkeys{Gin}{#1}%
434   \edef\mh@@repos{\mh@currentrepos}%
435   \ifx\Gin@mhrepos\empty%
436     \frameimage[#1]{\MathHub{\mh@currentrepos/source/#2}}%
437   \else%
438     \frameimage[#1]{\MathHub{\Gin@mhrepos/source/#2}}%
439   \fi%
440 }%
441 </mikoslides.ltxml | mikoslides>
442 <mikoslides.ltxml>');

```

4.8 problem-mh: Support for Problems

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

```

443 <*problem>
444 \ProvidesPackage{problem-mh}[2015/11/22 v1.0 MathHub support for the sTeX problem package]
445 \RequirePackage{mathhub}
446 </problem>
447 <*problem.ltxml>
448 \RequirePackage('mathhub');
449 </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`.

```

450 <*problem>
451 \newcommand\includemhproblem[2] [] {\metasetkeys{inclprob}{#1}%
452 \edef\mh@@repos{\mh@currentrepos}%
453 \ifx\inclprob\mhrepos\@empty\else\mhcurrentrepos\inclprob\mhrepos\fi%
454 \input{\MathHub{\mh@currentrepos/source/#2}}%
455 \mhcurrentrepos\mh@@repos\clear@inclprob@keys}
456 </problem>
457 <*problem.ltxml>
458 sub includemhproblem {
459   my ($gullet,$keyval,$arg2) = @_ ;
460   my $repo_path;
461   if ($keyval) {
462     $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
463   if (! $repo_path) {
464     $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }
465   else {
466     $keyval->setValue('mhrepos',undef); }
467   my $mathhub_base = ToString(Digest('\MathHub{'}));
468   my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);
469   return Invocation(T_CS('\includeproblem'), $keyval, T_OTHER($finalpath)); }#
470 DefKeyVal('inclprob','mhrepos','Semiverbatim');
471 DefMacro('\includemhproblem OptionalKeyVals:inclprob {}', \&includemhproblem);
472 </problem.ltxml>

```

4.9 hwexam-mh: Support for Assignments

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

```

473 <*hwexam>
474 \ProvidesPackage{hwexam-mh}[2015/11/22 v1.0 MathHub support for the sTeX hwexam package]
475 \RequirePackage{mathhub}
476 </hwexam>
477 <*hwexam.ltxml>
478 \RequirePackage('mathhub');
479 </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`.

```

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

```

`\inputmhassignment` analogous

```

503 <hwexam>
504 \newcommand\inputmhassignment[2][\metasetkeys{inclassig}{#1}%
505 \edef\mh@@repos{\mh@currentrepos}%
506 \ifx\inclassig@mhrepos\empty\else\mhcurrentrepos\inclassig@mhrepos\fi%
507 \inputassignment[#1]{\MathHub{\mh@currentrepos/source/#2}}%
508 \mhcurrentrepos\mh@@repos\clear@inclassig@keys}
509 </hwexam>
510 <hwexam.ltxml>
511 sub inputmhassignment {
512   my ($gullet,$keyval,$arg2) = @_;
513   my $repo_path;
514   if ($keyval) {
515     $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
516   if (! $repo_path) {
517     $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }
518   else {
519     $keyval->setValue('mhrepos',undef); }
520   my $mathhub_base = ToString(Digest('\MathHub{'}));
521   my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);
522   return Invocation(T_CS('\inputassignment'), $keyval, T_OTHER($finalpath)); }#$
523 DefMacro('\inputmhassignment OptionalKeyVals:inclprob {}', \&inputmhassignment);

```

524 \langle /hwexam.ltxml \rangle

4.10 tikzinput-mh: Support for Assignments

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

```
525  $\langle$ *tikzinput $\rangle$ 
526 \ProvidesPackage{tikzinput-mh}[2015/11/22 v1.0 MathHub support for the sTeX tikzinput package]
527 \RequirePackage{mathhub}
528  $\langle$ /tikzinput $\rangle$ 
529  $\langle$ *tikzinput.ltxml $\rangle$ 
530 \RequirePackage('mathhub');
531  $\langle$ /tikzinput.ltxml $\rangle$ 

532  $\langle$ tikzinput.ltxml $\rangle$ RawTeX('
533  $\langle$ *tikzinput | tikzinput.ltxml $\rangle$ 
534 \define@key{Gin}{mhrepos}{\csxdef\Gin@mhrepos{#1}}
535 \newcommand\mhtikzinput[2][\def\Gin@mhrepos{}\setkeys{Gin}{#1}%
536 \edef\mh@crepos{\mh@currentrepos}%
537 \ifx\Gin@mhrepos\empty\tikzinput[#1]{\MathHub{\mh@currentrepos/source/#2}}}%
538 \else\tikzinput[#1]{\MathHub{\Gin@mhrepos/source/#2}}\fi
539 \def\Gin@mhrepos{}\mhcurrentrepos\mh@crepos}
540 \newcommand\cmhtikzinput[2][\begin{center}\mhtikzinput[#1]{#2}\end{center}}
541  $\langle$ /tikzinput | tikzinput.ltxml $\rangle$ 
542  $\langle$ tikzinput.ltxml $\rangle$ ');
```

4.11 Finale

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

543 \langle ltxml | modules.ltxml | structview.ltxml | omtex.ltxml | statements.ltxml | smultiling.ltxml | mikoslides.ltxml | problem.

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 Kohlhasse. *metakeys.sty: A generic framework for extensible Metadata in L^AT_EX*. Tech. rep. Comprehensive T_EX 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).