

MathHub Support for \LaTeX^*

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

November 10, 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 2015/11/04)

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	18
4.9	<code>hwexam-mh</code> : Support for Assignments	19
4.10	Finale	21

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 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 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 omtext-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 `\endpackage`) and the \LaTeX ML bindings (between `\beginltxml` and `\endltxml`). 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 ML binding files and the base package.

```

1 \beginltxml | modules.ltxml | structview.ltxml | omtex.ltxml | statements.ltxml | smultiling.ltxml | mikosides.ltxml | problem
2 \package LaTeXML::Package::Pool;
3 use strict;
4 use LaTeXML::Package;
5 \endltxml | modules.ltxml | structview.ltxml | omtex.ltxml | statements.ltxml | smultiling.ltxml | mikosides.ltxml | problem
6 \package\ProvidesPackage{mathhub}[2015/11/04 v1.0 sTeX Support for MathHub.info]

7 \beginpackage
8 \DeclareOption*{}
9 \ProcessOptions
10 \endpackage
11 \beginltxml
12 \DeclareOption(undef,sub {});
13 \ProcessOptions();
14 \endltxml

```

Then we need to set up the packages by requiring the `metakeys` package [Koh15] to be loaded (in the right version).

```

15 \beginpackage
16 \RequirePackage{keyval}
17 \endpackage
18 \beginltxml
19 \RequirePackage('keyval');
20 \endltxml

```

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.

```

21 \beginpackage
22 \newcommand\mhcurrentrepos[1]{%
23   \edef\@test{#1}%
24   \ifx\@test\mhcurrentrepos% if new dir = old dir
25     \relax% no need to change
26   \else%
27     \protected@write\@auxout{}{\string\@mhcurrentrepos{#1}}%
28     \fi%
29     \@mhcurrentrepos{#1}% define mhcurrentrepos
30 }%
31 \newcommand\@mhcurrentrepos[1]{\edef\mhcurrentrepos{#1}}%

```

```

32 </package>
33 <*ltxml>
34 DefMacro('mhcurrentrepos{','\@mhcurrentrepos{#1}');
35 DefMacro('mhcurrentrepos{','\def\mh@currentrepos{#1}\@mhcurrentrepos{#1}');
36 DefConstructor('mhcurrentrepos{','',
37   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}
42 \newcommand\libinput[1]{\def\@libfile{\MathHub{\mh@currentrepos/lib/#1}}%
43 \IfFileExists{\@libfile}{\input\@libfile}%
44 {\edef\@group{\expandafter\modules@@first\mh@currentrepos;}
45 \edef\@infile{\MathHub{\@group/meta-inf/lib/#1}}
46 \IfFileExists{\@infile}{\input{\@infile}}%
47 {\PackageError{modules}
48   {Library file missing, cannot input #1\MessageBreak%
49     Both \@libfile.tex\MessageBreak and \@infile.tex\MessageBreak do not exit}%
50   {Check whether the file name is correct}}}%
51 </package | ltxml>
52 <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.

```

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{mathhub}
59 </modules>
60 <*modules.ltxml>
61 DeclareOption(undef,sub{PassOptions('modules','sty',ToString(Digest(T_CS('CurrentOption')))));
62 ProcessOptions();
63 RequirePackage('modules');
64 RequirePackage('mathhub');
65 </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.

```



```

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

```

```

116   importmoduleI($stomach,$whatsit);
117   AssignValue('current_repos' => $current_repos, 'global'); }
118   else {
119     importmoduleI($stomach,$whatsit); }
120   return; }
121
122 DefConstructor('\importMHmoduleI OptionalKeyVals:importmhmodule { }', '',
123   afterDigest=> \&importMHmoduleI );#$
124 </modules.ltxml>

```

and now the analogs

\usemhmodule

```

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

```

\mhinputref

```

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

```

```

160 </modules | modules.ltxml>
161 <modules.ltxml>');
\mhinput
162 <*modules>
163 \let\mhinput\mhinputref%
164 </modules>

```

4.3 omtex-mh: MH Variants for OMText

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

```

165 <*omtext>
166 \ProvidesPackage{omtex-mh}[2015/11/04 v1.0 MathHub support for the sTeX omtex package]
167 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{omtex}}
168 \ProcessOptions
169 \RequirePackage{mathhub}
170 \RequirePackage{omtex}
171 \RequirePackage{modules-mh}
172 </omtex>
173 <*omtex.ltxml>
174 \DeclareOption(undef,sub{PassOptions('omtex','sty',ToString(Digest(T_CS('\CurrentOption')))); }
175 \ProcessOptions();
176 \RequirePackage('mathhub');
177 \RequirePackage('omtex');
178 \RequirePackage('modules-mh');
179 </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.
180 <*omtex>
181 \addmetakey{Gin}{mhrepos}
182 \newcommand\mhgraphics[2][]{\metasetkeys{Gin}{#1}%
183 \edef\mh@crepos{\mh@currentrepos}%
184 \ifx\Gin@mhrepos\empty\mygraphics[#1]{\MathHub{\mh@currentrepos/source/#2}}%
185 \else\mygraphics[#1]{\MathHub{\Gin@mhrepos/source/#2}}\fi
186 \def\Gin@mhrepos{\mhcurrentrepos\mh@crepos}
187 \newcommand\mhgraphics[2][]{\begin{center}\mhgraphics[#1]{#2}\end{center}}
188 \newcommand\mhgraphics[2][]{\fbox{\mhgraphics[#1]{#2}}}
189 \newcommand\mhcbgraphics[2][]{\begin{center}\fbox{\mhgraphics[#1]{#2}}\end{center}}
190 </omtex>
191 <*omtex.ltxml>
192 sub mhgraphics {
193   my ($gullet,$keyval,$arg2) = @_ ;
194   my $repo_path;
195   if ($keyval) {
196     $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
197   if (! $repo_path) {
198     $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }

```

```

199 else {
200   $keyval->setValue('mhrepos',undef); }
201 my $mathhub_base = ToString(Digest('\MathHub{'}));
202 my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);
203 return Invocation(T_CS('@includegraphicx'), $keyval, T_OTHER($finalpath)); }#$
204 DefKeyVal('Gin','mhrepos','Semiverbatim');
205 DefMacro('\mhgraphics OptionalKeyVals:Gin {}', \&mhgraphics);
206 DefMacro('\mhcgraphics []{}', '\begin{center}\mhgraphics[#1]{#2}\end{center}');
207 DefMacro('\mhbgraphics []{}', '\fbox{\mhgraphics[#1]{#2}}');
208 </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.

```

209 <*statements>
210 \ProvidesPackage{statements-mh}[2015/11/04 v1.0 MathHub support for the sTeX statements package]
211 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{statements}}
212 \ProcessOptions
213 \RequirePackage{mathhub}
214 \RequirePackage{statements}
215 \RequirePackage{omtext-mh}
216 </statements>
217 <statements.ltxml>
218 DeclareOption(undef,sub{PassOptions('statements','sty',ToString(Digest(T_CS('\CurrentOption'))))}
219 ProcessOptions();
220 RequirePackage('mathhub');
221 RequirePackage('statements');
222 RequirePackage('omtext-mh');
223 </statements.ltxml>

224 <*statements>
225 \let\usemhvocab=\usemhmodule
226 </statements>
227 <statements.ltxml>
228 DefMacro('\usemhvocab','\usemhmodule');
229 </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.

```

230 <*smultiling>
231 \ProvidesPackage{smultiling-mh}[2015/11/04 v1.0 MathHub support for the sTeX smultiling package]
232 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{smultiling}}
233 \ProcessOptions
234 \RequirePackage{mathhub}
235 \RequirePackage{smultiling}
236 \RequirePackage{structview-mh}

```

```

237 </smultiling>
238 <*smultiling.ltxml>
239 DeclareOption(undef,sub{PassOptions('smultiling','sty',ToString(Digest(T_CS('\CurrentOption'))))
240 ProcessOptions();
241 RequirePackage('mathhub');
242 RequirePackage('smultiling');
243 RequirePackage('structview-mh');
244 </smultiling.ltxml>

mhmodnl:*
245 <*smultiling>
246 \addmetakey{mhmodnl}{repos}
247 \addmetakey{mhmodnl}{path}
248 \addmetakey*{mhmodnl}{title}
249 \addmetakey*{mhmodnl}{creators}
250 \addmetakey*{mhmodnl}{contributors}
251 \addmetakey{mhmodnl}{srccite}
252 \addmetakey{primary}{mhmodnl}[yes]
253 </smultiling>
254 <*smultiling.ltxml>
255 DefKeyVal('mhmodnl','title','Semiverbatim');
256 DefKeyVal('mhmodnl','repos','Semiverbatim');
257 DefKeyVal('mhmodnl','path','Semiverbatim');
258 DefKeyVal('mhmodnl','creators','Semiverbatim');
259 DefKeyVal('mhmodnl','contributors','Semiverbatim');
260 DefKeyVal('mhmodnl','primary','Semiverbatim');
261 </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.

262 <*smultiling>
263 \newenvironment{mhmodnl}[3][\metasetkeys{mhmodnl}{#1}%
264 \def\@test{#1}\ifx\@test\@empty\begin{module}[id=#2.#3]\else\begin{module}[id=#2.#3,#1]\fi%
265 \edef\@repos{\ifx\mhmodnl@repos\@empty\mh@currentrepos\else\mhmodnl@repos}
266 \if@langfiles\importmhmodule[repos=\@repos,load=#2,ext=tex]{#2}\else
267 \ifx\mhmodnl@load\@empty\importmodule{#2}\else\importmodule[ext=tex,load=\mhmodnl@load]{#2}\fi%
268 \fi}
269 {\end{module}}
270 </smultiling>
271 <*smultiling.ltxml>
272 DefEnvironment('{mhmodnl} OptionalKeyVals:mhmodnl {}{}',
273     "?#excluded()(<omdoc:theory xml:id='#2.#3' >"
274     . "??&defined(&GetKeyVal(#1,'creators'))(<dc:creator>&GetKeyVal(#1,'creators')</dc:cr
275     . "??&defined(&GetKeyVal(#1,'title'))(<dc:title>&GetKeyVal(#1,'title')</dc:title>())"
276     . "??&defined(&GetKeyVal(#1,'contributors'))(<dc:contributor>&GetKeyVal(#1,'contribut
277     . "<omdoc:imports from='?&GetKeyVal(#1,'load')(&canonical_omdoc_path(&GetKeyVal(#1,'
278     . "#body"
279     . "</omdoc:theory>)",
280     afterDigestBegin=>sub {
281     my ($stomach, $whatsit) = @_;

```

```

282 my $keyval = $whatsit->getArg(1);
283 my $signature = ToString($whatsit->getArg(2));
284 my $language = ToString($whatsit->getArg(3));
285 my $repos = ToString(GetKeyVal($keyval,'torepos'));
286 my $current_repos = LookupValue('current_repos');
287 if (!$repos) { $repos = $current_repos; }
288 my $defpaths = LookupValue('defpath');
289 my $load_path = ($$defpaths[MathHub]).$repos.'/source/'. $signature;
290
291 if ($keyval) {
292   # If we're not given load, AND the langfiles option is in effect,
293   # default to #2
294   if ((! $keyval->getValue('path')) && (LookupValue('smultiling_langfiles')) {
295     $keyval->setValue('load',$load_path); }
296   # Always load a TeX file
297   $keyval->setValue('ext','tex');
298   $keyval->setValue('id',"$signature.$language"); }
299 module_afterDigestBegin(@_);
300 importmoduleI(@_);
301 return; },
302 afterDigest=>sub {
303   module_afterDigest(@_); }));
304 </smultiling.ltxml>%$

```

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

```

305 <smultiling.ltxml>RawTeX('
306 <*smultiling | smultiling.ltxml>
307 \newenvironment{mhviewsig}[4] [] {\def@test{#1}\ifx@test\@empty%
308 \begin{mhview}[id=#2,ext=tex]{#3}{#4}\else%
309 \begin{mhview}[id=#2,#1,ext=tex]{#3}{#4}\fi}
310 {\end{mhview}}

```

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

```

311 \newenvironment{mhviewnl}[5] [] {\def@test{#1}\ifx@test\@empty%
312 \begin{mhviewsketch}[id=#2.#5,ext=tex]{#3}{#4}\else%
313 \begin{mhviewsketch}[id=#2.#5,#1,ext=tex]{#3}{#4}\fi}
314 {\end{mhviewsketch}}
315 </smultiling | smultiling.ltxml>
316 <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.

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)`!

```

317 <*structview>
318 \ProvidesPackage{structview-mh}[2015/11/04 v1.0 MathHub support for the sTeX structview package]
319 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{structview}}
320 \ProcessOptions
321 \RequirePackage{mathhub}
322 \RequirePackage{structview}
323 \RequirePackage{modules-mh}
324 </structview>
325 <*structview.ltxml>
326 DeclareOption(undef,sub{PassOptions('structview','sty',ToString(Digest(T_CS('\CurrentOption'))))
327 ProcessOptions();
328 RequirePackage('mathhub');
329 RequirePackage('structview');
330 RequirePackage('modules-mh');
331 </structview.ltxml>

```

importmhmodulevia

```

332 <structview.ltxml>RawTeX(
333 <*structview | structview.ltxml>
334 \newenvironment{importmhmodulevia}[3][[%
335 \gdef\@doit{\importmhmodule[#1]{#2}{#3}}%
336 \ifmod@show\par\noindent importing module #2 via \@doit\fi
337 ]{%
338 \aftergroup\@doit\ifmod@show end import\fi%
339 }%
340 </structview | structview.ltxml>
341 <structview.ltxml>');

342 <*structview>
343 \srefaddidkey{mhview}
344 \addmetakey{mhview}{display}
345 \addmetakey{mhview}{creators}
346 \addmetakey{mhview}{contributors}
347 \addmetakey{mhview}{srccite}
348 \addmetakey*{mhview}{title}
349 \addmetakey{mhview}{fromrepos}
350 \addmetakey{mhview}{torepos}
351 \addmetakey{mhview}{frompath}
352 \addmetakey{mhview}{topath}
353 \addmetakey[sms]{mhview}{ext}
354 </structview>
355 <*structview.ltxml>
356 DefKeyVal('mhview','id','Semiverbatim');
357 DefKeyVal('mhview','display','Semiverbatim');
358 DefKeyVal('mhview','creators','Semiverbatim');
359 DefKeyVal('mhview','contributors','Semiverbatim');
360 DefKeyVal('mhview','srccite','Semiverbatim');
361 DefKeyVal('mhview','title','Semiverbatim');
362 DefKeyVal('mhview','fromrepos','Semiverbatim');
363 DefKeyVal('mhview','torepos','Semiverbatim');

```

```

364 DefKeyVal('mhview','frompath','Semiverbatim');
365 DefKeyVal('mhview','topath','Semiverbatim');
366 DefKeyVal('mhview','ext','Semiverbatim');
367 </structview.ltxml>

```

mhview the MathHub version

```

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

```

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

```

405 <*structview>
406 \newenvironment{@mhview}[2][{}]{%from, to
407 \importmhmodule[repos=\mhview@fromrepos,path=\mhview@frompath,ext=\mhview@ext]{#1}%
408 \importmhmodule[repos=\mhview@torepos,path=\mhview@topath,ext=\mhview@ext]{#2}%
409 }{}%

```



```
410 </structview>
```

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

```
411 <*structview>
412 \newenvironment{mhviewsketch}[3][{}]{%
413   \metasetkeys{mhview}{#1}%
414   \sref@target%
415   \begin{@mhview}{#2}{#3}%
416   \view@heading{#2}{#3}{\mhview@display}{\mhview@title}%
417 }{%
418   \end{@mhview}%
419   \ignorespaces%
420 }%
421 \ifmod@show\surroundwithmdframed{mhviewsketch}\fi
422 </structview>
423 <*structview.ltxml>
424 DefMacroI(T_CS('\begin{mhviewsketch}'), 'OptionalKeyVals:mhview {}{}', sub {
425   my ($gullet, $keyvals, $from_arg, $to_arg) = @_;
426   my $from = ToString(Digest($from_arg));
427   my $to = ToString(Digest($to_arg));
428   my $from_repos = ToString(GetKeyVal($keyvals, 'fromrepos'));
429   my $to_repos = ToString(GetKeyVal($keyvals, 'torepos'));
430   my $repos = LookupValue('current_repos');
431   my $from_path = ToString(GetKeyVal($keyvals, 'frompath'));
432   my $to_path = ToString(GetKeyVal($keyvals, 'topath'));
433   my $ext = ToString(GetKeyVal($keyvals, 'ext')) if $keyvals;
434   $ext = 'sms' unless $ext;
435   my $current_repos = LookupValue('current_repos');
436   if (!$from_repos) { $from_repos = $current_repos; }
437   if (!$to_repos) { $to_repos = $current_repos; }
438   return (
439     Tokenize("\importMHmoduleI[repos=$from_repos,path=$from_path,ext=$ext]{$from}")->unlist,
440     Tokenize("\importMHmoduleI[repos=$to_repos,path=$to_path,ext=$ext]{$to}")->unlist,
441     Invocation(T_CS('\begin{viewsketchenv}'), $keyvals, $from_arg, $to_arg)->unlist
442   );
443 });
444 DefMacroI('\end{mhviewsketch}', undef, '\end{viewsketchenv}');
445 </structview.ltxml>
```

4.7 mikosides-mh: Support for MiKo Slides

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

```
446 <*mikosides>
447 \ProvidesPackage{mikosides-mh}[2015/11/04 v1.0 MathHub support for the sTeX mikosides package]
448 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{mikosides}}
449 \ProcessOptions
450 \RequirePackage{mathhub}
451 \RequirePackage{mikosides}
```

```

452 \RequirePackage{statements-mh}
453 \</mikoslides>
454 \<*mikoslides.ltxml>
455 DeclareOption(undef,sub{PassOptions('mikoslides','sty',ToString(Digest(T_CS('\CurrentOption'))))
456 ProcessOptions();
457 RequirePackage('mathhub');
458 RequirePackage('mikoslides');
459 RequirePackage('statements-mh');
460 \</mikoslides.ltxml>

\mhframeimage Use the current value of \mh@currentrepos or the value of the mhrepos key if it
is given in \frameimage.
461 \<mikoslides>\addmetakey{Gin}{mhrepos}
462 \<mikoslides.ltxml>\DefKeyVal('Gin','mhrepos','Semiverbatim');
463 \<mikoslides.ltxml>\RawTeX('
464 \<*mikoslides.ltxml | mikoslides>
465 \newcommand\mhframeimage[2][{}]{%
466 \metasetkeys{Gin}{#1}%
467 \edef\mh@@repos{\mh@currentrepos}%
468 \ifx\Gin\mhrepos\@empty%
469 \frameimage[#1]{\MathHub{\mh@currentrepos/source/#2}}%
470 \else%
471 \frameimage[#1]{\MathHub{\Gin\mhrepos/source/#2}}%
472 \fi%
473 }%
474 \</mikoslides.ltxml | mikoslides>
475 \<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.

```

476 \<*problem>
477 \ProvidesPackage{problem-mh}[2015/11/04 v1.0 MathHub support for the sTeX problem package]
478 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{problem}}
479 \ProcessOptions
480 \RequirePackage{mathhub}
481 \RequirePackage{problem}
482 \RequirePackage{omtext-mh}
483 \</problem>
484 \<*problem.ltxml>
485 DeclareOption(undef,sub{PassOptions('problem','sty',ToString(Digest(T_CS('\CurrentOption'))))});
486 ProcessOptions();
487 RequirePackage('mathhub');
488 RequirePackage('problem');
489 RequirePackage('omtext-mh');
490 \</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`.

```

491 <*problem>
492 \newcommand\includemhproblem[2][\metasetkeys{inclprob}{#1}%
493 \edef\mh@@repos{\mh@currentrepos}%
494 \ifx\inclprob@mhrepos\empty\else\mhcurrentrepos\inclprob@mhrepos\fi%
495 \input{\MathHub{\mh@currentrepos/source/#2}}%
496 \mhcurrentrepos\mh@@repos\clear@inclprob@keys}
497 </problem>
498 <*problem.ltxml>
499 sub includemhproblem {
500   my ($gullet,$keyval,$arg2) = @_ ;
501   my $repo_path;
502   if ($keyval) {
503     $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
504   if (! $repo_path) {
505     $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }
506   else {
507     $keyval->setValue('mhrepos',undef); }
508   my $mathhub_base = ToString(Digest('\MathHub{ }'));
509   my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);
510   return Invocation(T_CS('\includeproblem'), $keyval, T_OTHER($finalpath)); }#$
511 DefKeyVal('inclprob','mhrepos','Semiverbatim');
512 DefMacro('\includemhproblem OptionalKeyVals:inclprob { }', \&includemhproblem);
513 </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.

```

514 <*hwexam>
515 \ProvidesPackage{hwexam-mh}[2015/11/04 v1.0 MathHub support for the sTeX hwexam package]
516 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{hwexam}}
517 \ProcessOptions
518 \RequirePackage{mathhub}
519 \RequirePackage{hwexam}
520 \RequirePackage{problem-mh}
521 </hwexam>
522 <*hwexam.ltxml>
523 DeclareOption(undef,sub{PassOptions('hwexam','sty',ToString(Digest(T_CS('\CurrentOption')))); }
524 ProcessOptions();
525 RequirePackage('mathhub');
526 RequirePackage('hwexam');
527 RequirePackage('problem-mh');
528 </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`.

```

529 <hwexam>
530 \newcommand\includemhassignment[2][\metasetkeys{inclassig}{#1}%
531 \edef\mh@@repos{\mh@currentrepos}%
532 \ifx\inclassig@mhrepos\empty\else\mhcurrentrepos\inclassig@mhrepos\fi%
533 \includeassignment[#1]{\MathHub{\mh@currentrepos/source/#2}}%
534 \mhcurrentrepos\mh@@repos\clear@inclassig@keys}
535 </hwexam>
536 <hwexam.ltxml>
537 sub includemhassignment {
538   my ($gullet,$keyval,$arg2) = @_;
539   my $repo_path;
540   if ($keyval) {
541     $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
542   if (! $repo_path) {
543     $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }
544   else {
545     $keyval->setValue('mhrepos',undef); }
546   my $mathhub_base = ToString(Digest('\MathHub{ }'));
547   my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);
548   return Invocation(T_CS('\includeassignment'), $keyval, T_OTHER($finalpath)); }#$
549 DefKeyVal('inclprob','mhrepos','Semiverbatim');
550 DefMacro('\includemhassignment OptionalKeyVals:inclprob { }', \&includemhassignment);
551 </hwexam.ltxml>

```

`\inputmhassignment` analogous

```

552 <hwexam>
553 \newcommand\inputmhassignment[2][\metasetkeys{inclassig}{#1}%
554 \edef\mh@@repos{\mh@currentrepos}%
555 \ifx\inclassig@mhrepos\empty\else\mhcurrentrepos\inclassig@mhrepos\fi%
556 \inputassignment[#1]{\MathHub{\mh@currentrepos/source/#2}}%
557 \mhcurrentrepos\mh@@repos\clear@inclassig@keys}
558 </hwexam>
559 <hwexam.ltxml>
560 sub inputmhassignment {
561   my ($gullet,$keyval,$arg2) = @_;
562   my $repo_path;
563   if ($keyval) {
564     $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
565   if (! $repo_path) {
566     $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }
567   else {
568     $keyval->setValue('mhrepos',undef); }
569   my $mathhub_base = ToString(Digest('\MathHub{ }'));
570   my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);
571   return Invocation(T_CS('\inputassignment'), $keyval, T_OTHER($finalpath)); }#$
572 DefMacro('\inputmhassignment OptionalKeyVals:inclprob { }', \&inputmhassignment);
573 </hwexam.ltxml>

```

4.10 Finale

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

```
574 <ltxml | modules.ltxml | structview.ltxml | omtext.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).