

# MathHub Support for $\text{\LaTeX}^*$

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

November 12, 2015

## Abstract

The `sref` package is part of the  $\text{\LaTeX}$  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  $\text{\LaTeX}$  with support for the MathHub.info portal

## Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>The User Interface</b>	<b>3</b>
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
<b>3</b>	<b>Limitations</b>	<b>6</b>
<b>4</b>	<b>Implementation</b>	<b>7</b>
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 . . . . .	13
4.6	<code>structview-mh</code> : MH Variants for Structures and Views . . . . .	15
4.7	<code>mikoslides-mh</code> : Support for MiKo Slides . . . . .	18
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  $\text{\LaTeX}$  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 [HorIacJuc:cscpnrr11]. But this structure can be hidden from the  $\text{\LaTeX}$  author with **MathHub**-enabled versions of the  $\text{\LaTeX}$  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  $\text{\LaTeX}$  nor  $\text{\LaTeXML}$  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  $\text{\LaTeX}$  `\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}
```

---

<sup>1</sup>EDNOTE: needs to be documented

<sup>2</sup>EDNOTE: mhmodsig seems to be missing what happened?

<sup>3</sup>EDNOTE: 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:github:on`].

1. none reported yet.

## 4 Implementation

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

```

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

8 <*package>
9 \DeclareOption*{}
10 \ProcessOptions
11 </package>
12 <*ltxml>
13 use LaTeXML::Util::Pathname;
14 \DeclareOption(undef,sub {});
15 \ProcessOptions();
16 </ltxml>
```

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

```

17 <*package>
18 \RequirePackage{keyval}
19 </package>
20 <*ltxml>
21 \RequirePackage('keyval');
22 </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.

```

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

```

32 }%
33 \newcommand\mhcurrentrepos[1]{\edef\mh@currentrepos{#1}}%
34 \end{package}
35 \end{*xml}
36 DefMacro('mhcurrentrepos{','}\mhcurrentrepos{#1}');
37 DefMacro('mhcurrentrepos{','}\def\mh@currentrepos{#1}\mhcurrentrepos{#1}');
38 DefConstructor('mhcurrentrepos{','}',
39 afterDigest => sub{ AssignValue('current_repos',ToString($_[1]->getArg(1)),'global'); } );
40 \end{*xml}#\$

```

`\libinput` the `\libinput` macro inputs from the `lib` directory of the MathHub repository or the `meta-inf/lib` repos of the group.

```

41 \end{package}
42 \def\modules@@first#1/#2;{#1}
43 \newcommand\libinput[1]{\def\@libfile{\MathHub\mh@currentrepos/lib/#1}}%
44 \IfFileExists{\@libfile}{\input\@libfile}%
45 {\edef\@group{\expandafter\modules@@first\mh@currentrepos;}
46 \edef\@inffile{\MathHub\@group/meta-inf/lib/#1}
47 \IfFileExists{\@inffile}{\input\@inffile}}%
48 {\PackageError{modules}
49 {Library file missing, cannot input #1\MessageBreak%
50 Both \@libfile.tex\MessageBreak and \@inffile.tex\MessageBreak do not exist}%
51 {Check whether the file name is correct}}}%
52 \end{package}
53 \end{*xml}
54 DefMacro('modules@@first#1/#2;','\#1');
55 DefMacro('libinput {','}', sub{
56 my ($gullet, $name) = @_;
57 $name = ToString($name);
58 #Relative paths for recursive search
59 my $FIRSTLIB = ('/../../lib');
60 my $SECONDLIB = ('/../../meta-info/lib');
61 my $file = pathname_find($name, types => ['tex'], paths => [$FIRSTLIB]);
62 $file = pathname_find($name, types=>['tex'], paths=>[$SECONDLIB]) unless $file;
63 # Singal error if the file cannot be found
64 LaTeXML::Package::InputContent($file, noerror=>1); });
65 \end{*xml}

```

## 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.

```

66 \end{modules}
67 \ProvidesPackage{modules-mh}[2015/11/04 v1.0 MathHub support for the sTeX modules package]
68 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{modules}}
69 \ProcessOptions
70 \RequirePackage{modules}
71 \RequirePackage{mathhub}
72 \end{modules}

```



```

73 <*modules.ltxml>
74 DeclareOption(undef,sub{PassOptions('modules','sty',ToString(Digest(T_CS('\CurrentOption'))));
75 ProcessOptions();
76 RequirePackage('modules');
77 RequirePackage('mathhub');
78 </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.

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

```

```

115 sub importMHmoduleI {
116   my ($stomach, $whatsit) = @_;
117   my $keyval = $whatsit->getArg(1);
118   my $id = $whatsit->getArg(2);
119   if ($keyval) {
120     my $repos = ToString($keyval->getValue('repos'));
121     my $path = ToString($keyval->getValue('path'));
122     my $current_repos = LookupValue('current_repos');
123     if (!$repos) { # Use the implicit current repository
124       $repos = $current_repos; }
125     my $defpaths = LookupValue('defpath');
126     my $load_path = ($$defpaths{MathHub}).$repos.'/source/'. $path;
127     $keyval->setValue('load', $load_path);
128     AssignValue('current_repos' => $repos, 'global');
129     importmoduleI($stomach, $whatsit);
130     AssignValue('current_repos' => $current_repos, 'global'); }
131   else {
132     importmoduleI($stomach, $whatsit); }
133   return; }
134
135 DefConstructor('\importMHmoduleI OptionalKeyVals:importmhmodule {}', '',
136   afterDigest=> \&importMHmoduleI );#$
137 </modules.ltxml>

```

and now the analogs

\usemhmodule

```

138 <*modules>
139 \newcommand\usemhmodule[2][]{%
140   \metasetkeys{importmhmodule}{#1}%
141   \ifx\importmhmodule@path\@empty%
142     \usemodule[ext=\importmhmodule@ext,id=\importmhmodule@id]{#2}%
143   \else%
144     \edef\mh@@repos{\mh@currentrepos}%
145     \ifx\importmhmodule@repos\@empty%
146       \else%
147         \mhcurrentrepos{\importmhmodule@repos}%
148       \fi%
149     \usemodule[load=\MathHub{\mh@currentrepos/source/\importmhmodule@path},ext=\importmhmodule@
150     \mhcurrentrepos\mh@@repos%
151   \fi%
152   \ignorespaces%
153 }%
154 </modules>
155 <*modules.ltxml>
156 DefConstructor('\usemhmodule OptionalKeyVals:importmhmodule {}',
157   "<omdoc:uses from='?&GetKeyVal(#1,'load')(&canonical_omdoc_path(&GetKeyVal(#1,'load')))(###
158   afterDigest => \&importMHmoduleI);
159 </modules.ltxml>

```

```

\mhinputref
160 <modules.ltxml>RawTeX( '
161 <*modules | modules.ltxml>
162 \newcommand\mhinputref[2] [] {%
163   \def\@repos{#1}%
164   \edef\mh@crepos{\mh@currentrepos}%
165   \ifx\@repos\empty%
166     \else%
167       \mhcurrentrepos{#1}%
168     \fi%
169   \inputref{\MathHub{\mh@currentrepos/source/#2}}%
170   \mhcurrentrepos\mh@crepos%
171   \ignorespaces%
172 }%
173 </modules | modules.ltxml>
174 <modules.ltxml>' );

```

```

\mhinput
175 <*modules>
176 \let\mhinput\mhinputref%
177 </modules>

```

### 4.3 omtex-mh: MH Variants for OMText

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

```

178 <*omtext>
179 \ProvidesPackage{omtext-mh}[2015/11/04 v1.0 MathHub support for the sTeX omtext package]
180 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{omtext}}
181 \ProcessOptions
182 \RequirePackage{mathhub}
183 \RequirePackage{omtext}
184 \RequirePackage{modules-mh}
185 </omtext>
186 <*omtext.ltxml>
187 \DeclareOption{undef,sub{PassOptions('omtext','sty',ToString(Digest(T_CS('\CurrentOption')))); }
188 \ProcessOptions();
189 \RequirePackage('mathhub');
190 \RequirePackage('omtext');
191 \RequirePackage('modules-mh');
192 </omtext.ltxml>

```

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

```

193 <*omtext>
194 \addmetakey{Gin}{mhrepos}
195 \newcommand\mhgraphics[2] [] {\metasetkeys{Gin}{#1}%
196 \edef\mh@crepos{\mh@currentrepos}%

```

```

197 \ifx\Gin@mhrepos\empty\mygraphics[#1]{\MathHub{\mh@currentrepos/source/#2}}%
198 \else\mygraphics[#1]{\MathHub{\Gin@mhrepos/source/#2}}\fi
199 \def\Gin@mhrepos{\mhcurrentrepos\mh@currentrepos}
200 \newcommand\mhgraphics[2][\begin{center}\mhgraphics[#1]{#2}\end{center}]
201 \newcommand\mhgraphics[2][\fbox{\mhgraphics[#1]{#2}}]
202 \newcommand\mhcbgraphics[2][\begin{center}\fbox{\mhgraphics[#1]{#2}}\end{center}]
203 \</omtext>
204 \<omtext.ltxml>
205 sub mhgraphics {
206   my ($gullet,$keyval,$arg2) = @_;
207   my $repo_path;
208   if ($keyval) {
209     $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
210   if (! $repo_path) {
211     $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }
212   else {
213     $keyval->setValue('mhrepos',undef); }
214   my $mathhub_base = ToString(Digest('\MathHub{}'));
215   my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);
216   return Invocation(T_CS('@includegraphicx'), $keyval, T_OTHER($finalpath)); }#$
217 DefKeyVal('Gin','mhrepos','Semiverbatim');
218 DefMacro('\mhgraphics OptionalKeyVals:Gin {}', \&mhgraphics);
219 DefMacro('\mhgraphics []{}', '\begin{center}\mhgraphics[#1]{#2}\end{center}');
220 DefMacro('\mhcbgraphics []{}', '\fbox{\mhgraphics[#1]{#2}}');
221 \</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.

```

222 \<statements>
223 \ProvidesPackage{statements-mh}[2015/11/04 v1.0 MathHub support for the sTeX statements package]
224 \DeclareOption*\PassOptionsToPackage{\CurrentOption}{statements}
225 \ProcessOptions
226 \RequirePackage{mathhub}
227 \RequirePackage{statements}
228 \RequirePackage{omtext-mh}
229 \</statements>
230 \<statements.ltxml>
231 DeclareOption(undef,sub{PassOptions('statements','sty',ToString(Digest(T_CS('\CurrentOption'))))}
232 ProcessOptions();
233 RequirePackage('mathhub');
234 RequirePackage('statements');
235 RequirePackage('omtext-mh');
236 \</statements.ltxml>

237 \<statements>
238 \let\usemhvocab=\usemhmodule
239 \</statements>
240 \<statements.ltxml>

```

```

241 DefMacro('usemhvocab','usemhmodule');
242 \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.

```

243 \smultiling
244 \ProvidesPackage{smultiling-mh}[2015/11/04 v1.0 MathHub support for the sTeX smultiling package]
245 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{smultiling}}
246 \ProcessOptions
247 \RequirePackage{mathhub}
248 \RequirePackage{smultiling}
249 \RequirePackage{structview-mh}
250 \smultiling
251 \smultiling.ltxml
252 DeclareOption(undef,sub{PassOptions('smultiling','sty',ToString(Digest(T_CS('CurrentOption'))))})
253 ProcessOptions();
254 RequirePackage('mathhub');
255 RequirePackage('smultiling');
256 RequirePackage('structview-mh');
257 \smultiling.ltxml

```

`mhmodnl:`

```

258 \smultiling
259 \addmetakey{mhmodnl}{repos}
260 \addmetakey{mhmodnl}{path}
261 \addmetakey*{mhmodnl}{title}
262 \addmetakey*{mhmodnl}{creators}
263 \addmetakey*{mhmodnl}{contributors}
264 \addmetakey{mhmodnl}{srccite}
265 \addmetakey{primary}{mhmodnl}[yes]
266 \smultiling
267 \smultiling.ltxml
268 DefKeyVal('mhmodnl','title','Semiverbatim');
269 DefKeyVal('mhmodnl','repos','Semiverbatim');
270 DefKeyVal('mhmodnl','path','Semiverbatim');
271 DefKeyVal('mhmodnl','creators','Semiverbatim');
272 DefKeyVal('mhmodnl','contributors','Semiverbatim');
273 DefKeyVal('mhmodnl','primary','Semiverbatim');
274 \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.

```

275 \smultiling
276 \newenvironment{mhmodnl}[3][\metasetkeys{mhmodnl}{#1}%
277 \def\@test{#1}\ifx\@test\@empty\begin{module}[id=#2.#3]\else\begin{module}[id=#2.#3,#1]\fi%
278 \edef\@repos{\ifx\mhmodnl@repos\@empty\mh@currentrepos\else\mhmodnl@repos}
279 \if@langfiles\importmhmodule[repos=\@repos,load=#2,ext=tex]{#2}\else

```

```

280 \ifx\mhmodnl@load\empty\importmodule{#2}\else\importmodule[ext=tex,load=\mhmodnl@load]{#2}\fi%
281 \fi}
282 {\end{module}}
283 \smultiling
284 \smultiling.ltxml)
285 DefEnvironment('{mhmodnl} OptionalKeyVals:mhmodnl {}{}',
286     "?#excluded()(<omdoc:theory xml:id='#2.#3' >"
287     .   "?&defined(&GetKeyVal(#1,'creators'))(<dc:creator>&GetKeyVal(#1,'creators')</dc:cr
288     .   "?&defined(&GetKeyVal(#1,'title'))(<dc:title>&GetKeyVal(#1,'title')</dc:title>())"
289     .   "?&defined(&GetKeyVal(#1,'contributors'))(<dc:contributor>&GetKeyVal(#1,'contribut
290     .   "<omdoc:imports from='?&GetKeyVal(#1,'load')(&canonical_omdoc_path(&GetKeyVal(#1,'
291     .   "#body"
292     .   "</omdoc:theory>)",
293     afterDigestBegin=>sub {
294     my ($stomach, $whatsit) = @_;
295     my $keyval = $whatsit->getArg(1);
296     my $signature = ToString($whatsit->getArg(2));
297     my $language = ToString($whatsit->getArg(3));
298     my $repos = ToString(GetKeyVal($keyval,'torepos'));
299     my $current_repos = LookupValue('current_repos');
300     if (!$repos) { $repos = $current_repos; }
301     my $defpaths = LookupValue('defpath');
302     my $load_path = ($$defpaths[MathHub]).$repos.'/source/'. $signature;
303
304     if ($keyval) {
305     # If we're not given load, AND the langfiles option is in effect,
306     # default to #2
307     if ((! $keyval->getValue('path')) && (LookupValue('smultiling_langfiles'))){
308     $keyval->setValue('load',$load_path); }
309     # Always load a TeX file
310     $keyval->setValue('ext','tex');
311     $keyval->setValue('id',"$signature.$language"); }
312     module_afterDigestBegin(@_);
313     importmoduleI(@_);
314     return; },
315     afterDigest=>sub {
316     module_afterDigest(@_); });
317 \smultiling.ltxml)%$

```

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

```

318 \smultiling.ltxml)RawTeX(
319 \smultiling | smultiling.ltxml)
320 \newenvironment{mhviewsig}[4][[]]{\def\@test{#1}\ifx\@test\empty%
321 \begin{mhview}[id=#2,ext=tex]{#3}{#4}\else%
322 \begin{mhview}[id=#2,#1,ext=tex]{#3}{#4}\fi}
323 {\end{mhview}}

```

**mhviewnl** The mhviewnl environment is just a layer over the mhviewsketch environment

EdN:4

with the keys and language suitably adapted.<sup>4</sup>

```
324 \newenvironment{mhviewnl}[5] [] {\def@test{#1}\ifx@test\@empty%
325 \begin{mhviewsketch}[id=#2.#5,ext=tex]{#3}{#4}\else%
326 \begin{mhviewsketch}[id=#2.#5,#1,ext=tex]{#3}{#4}\fi}
327 {\end{mhviewsketch}}
328 \</smultiling | smultiling.ltxml>
329 \<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.

```
330 \<*structview>
331 \ProvidesPackage{structview-mh}[2015/11/04 v1.0 MathHub support for the sTeX structview package]
332 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{structview}}
333 \ProcessOptions
334 \RequirePackage{mathhub}
335 \RequirePackage{structview}
336 \RequirePackage{modules-mh}
337 \</structview>
338 \<*structview.ltxml>
339 \DeclareOption{undef,sub{PassOptions('structview','sty',ToString(Digest(T_CS('\CurrentOption'))))}}
340 \ProcessOptions();
341 \RequirePackage('mathhub');
342 \RequirePackage('structview');
343 \RequirePackage('modules-mh');
344 \</structview.ltxml>
```

`importmhmodulevia`

```
345 \<structview.ltxml>\RawTeX('
346 \<*structview | structview.ltxml>
347 \newenvironment{importmhmodulevia}[3] [] {%
348   \gdef\@@doit{\importmhmodule[#1]{#2}{#3}}%
349   \ifmod@show\par\noindent importing module #2 via \@@doit\fi
350 }{%
351   \aftergroup\@@doit\ifmod@show end import\fi%
352 }%
353 \</structview | structview.ltxml>
354 \<structview.ltxml>');;
```

  

```
355 \<*structview>
356 \srefaddidkey{mhview}
357 \addmetakey{mhview}{display}
358 \addmetakey{mhview}{creators}
359 \addmetakey{mhview}{contributors}
360 \addmetakey{mhview}{srccite}
```

---

<sup>4</sup>EdNOTE: 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`!

```

361 \addmetakey*{mhview}{title}
362 \addmetakey{mhview}{fromrepos}
363 \addmetakey{mhview}{torepos}
364 \addmetakey{mhview}{frompath}
365 \addmetakey{mhview}{topath}
366 \addmetakey[sms]{mhview}{ext}
367 \</structview>
368 \<*structview.ltxml>
369 DefKeyVal('mhview','id','Semiverbatim');
370 DefKeyVal('mhview','display','Semiverbatim');
371 DefKeyVal('mhview','creators','Semiverbatim');
372 DefKeyVal('mhview','contributors','Semiverbatim');
373 DefKeyVal('mhview','srccite','Semiverbatim');
374 DefKeyVal('mhview','title','Semiverbatim');
375 DefKeyVal('mhview','fromrepos','Semiverbatim');
376 DefKeyVal('mhview','torepos','Semiverbatim');
377 DefKeyVal('mhview','frompath','Semiverbatim');
378 DefKeyVal('mhview','topath','Semiverbatim');
379 DefKeyVal('mhview','ext','Semiverbatim');
380 \</structview.ltxml>

```

**mhview** the MathHub version

```

381 \<*structview>
382 \newenvironment{mhview}[3][\% keys, from, to
383 \metasetkeys{mhview}{#1}%
384 \sref@target%
385 \begin{@mhview}{#2}{#3}%
386 \view@heading{#2}{#3}{\mhview@display}{\mhview@title}%
387 }{\%
388 \end{@mhview}%
389 \ignorespaces%
390 }%
391 \ifmod@show\surroundwithmdframed{mhview}\fi
392 \</structview>
393 \<*structview.ltxml>
394 DefMacroI(T_CS('\begin{mhview}'),'OptionalKeyVals:mhview {}{}', sub {
395 my ($gullet, $keyvals, $from_arg, $to_arg) = @_;
396 my $from = ToString(Digest($from_arg));
397 my $to = ToString(Digest($to_arg));
398 AssignValue(from_module => $from);
399 AssignValue(to_module => $to);
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]{$to}")->unlist,
413   Invocation(T_CS('\begin{viewenv}'),$keyvals,$from_arg,$to_arg)->unlist
414 );
415 });
416 DefMacroI('\end{mhview}',undef,'\end{viewenv}');
417 </structview.ltxml>

```

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

```

418 <*structview>
419 \newenvironment{@mhview}[2]{%from, to
420   \importmhmodule[repos=\mhview@fromrepos,path=\mhview@frompath,ext=\mhview@ext]{#1}%
421   \importmhmodule[repos=\mhview@torepos,path=\mhview@topath,ext=\mhview@ext]{#2}%
422 }{}%
423 </structview>

```

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

```

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

```

```

455   );
456   });
457   DefMacroI('\end{mhviewsketch}',undef,'\end{viewsketchenv}');
458   </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.

```

459 <*mikoslides>
460 \ProvidesPackage{mikoslides-mh}[2015/11/04 v1.0 MathHub support for the sTeX mikoslides package]
461 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{mikoslides}}
462 \ProcessOptions
463 \RequirePackage{mathhub}
464 \RequirePackage{mikoslides}
465 \RequirePackage{statements-mh}
466 </mikoslides>
467 <*mikoslides.ltxml>
468 DeclareOption(undef,sub{PassOptions('mikoslides','sty',ToString(Digest(T_CS('\CurrentOption'))))}
469 ProcessOptions();
470 RequirePackage('mathhub');
471 RequirePackage('mikoslides');
472 RequirePackage('statements-mh');
473 </mikoslides.ltxml>

```

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

```

474 <mikoslides>\addmetakey{Gin}{mhrepos}
475 <mikoslides.ltxml>DefKeyVal('Gin','mhrepos','Semiverbatim');
476 <mikoslides.ltxml>RawTeX('
477 <*mikoslides.ltxml | mikoslides>
478 \newcommand\mhframeimage[2][{}]{%
479   \metasetkeys{Gin}{#1}%
480   \edef\mh@@repos{\mh@currentrepos}%
481   \ifx\Gin@mhrepos\@empty%
482     \frameimage[#1]{\MathHub{\mh@currentrepos/source/#2}}%
483   \else%
484     \frameimage[#1]{\MathHub{\Gin@mhrepos/source/#2}}%
485   \fi%
486 }%
487 </mikoslides.ltxml | mikoslides>
488 <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.

```

489 <*problem>

```

```

490 \ProvidesPackage{problem-mh}[2015/11/04 v1.0 MathHub support for the sTeX problem package]
491 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{problem}}
492 \ProcessOptions
493 \RequirePackage{mathhub}
494 \RequirePackage{problem}
495 \RequirePackage{omtext-mh}
496 \</problem>
497 \<*problem.ltxml>
498 \DeclareOption(undef,sub{PassOptions('problem','sty',ToString(Digest(T_CS('\CurrentOption'))));
499 \ProcessOptions();
500 \RequirePackage('mathhub');
501 \RequirePackage('problem');
502 \RequirePackage('omtext-mh');
503 \</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`.

```

504 \<*problem>
505 \newcommand\includemhproblem[2][\metasetkeys{inclprob}]{#1}%
506 \edef\mh@@repos{\mh@currentrepos}%
507 \ifx\inclprob\mhrepos\empty\else\mhcurrentrepos\inclprob\mhrepos\fi%
508 \input{\MathHub{\mh@currentrepos/source/#2}}%
509 \mhcurrentrepos\mh@@repos\clear@inclprob@keys}
510 \</problem>
511 \<*problem.ltxml>
512 sub includemhproblem {
513   my ($gullet,$keyval,$arg2) = @_ ;
514   my $repo_path;
515   if ($keyval) {
516     $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
517   if (! $repo_path) {
518     $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }
519   else {
520     $keyval->setValue('mhrepos',undef); }
521   my $mathhub_base = ToString(Digest('\MathHub{'}));
522   my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);
523   return Invocation(T_CS('\includeproblem'), $keyval, T_OTHER($finalpath)); }##$
524 DefKeyVal('inclprob','mhrepos','Semiverbatim');
525 DefMacro('\includemhproblem OptionalKeyVals:inclprob {}', \&includemhproblem);
526 \</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.

```

527 \<*hwexam>
528 \ProvidesPackage{hwexam-mh}[2015/11/04 v1.0 MathHub support for the sTeX hwexam package]

```

```

529 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{hwexam}}
530 \ProcessOptions
531 \RequirePackage{mathhub}
532 \RequirePackage{hwexam}
533 \RequirePackage{problem-mh}
534 \</hwexam>
535 \<hwexam.ltxml>
536 \DeclareOption(undef,sub{PassOptions('hwexam','sty',ToString(Digest(T_CS('\CurrentOption')))); }
537 \ProcessOptions();
538 \RequirePackage('mathhub');
539 \RequirePackage('hwexam');
540 \RequirePackage('problem-mh');
541 \</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`.

```

542 \<hwexam>
543 \newcommand\includemhassignment[2][\metasetkeys{inclassig}{#1}%
544 \edef\mh@@repos{\mh@currentrepos}%
545 \ifx\inclassig@mhrepos\empty\else\mhcurrentrepos\inclassig@mhrepos\fi%
546 \includeassignment[#1]{\MathHub{\mh@currentrepos/source/#2}}%
547 \mhcurrentrepos\mh@@repos\clear@inclassig@keys}
548 \</hwexam>
549 \<hwexam.ltxml>
550 sub includemhassignment {
551   my ($gullet,$keyval,$arg2) = @_;
552   my $repo_path;
553   if ($keyval) {
554     $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
555   if (! $repo_path) {
556     $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }
557   else {
558     $keyval->setValue('mhrepos',undef); }
559   my $mathhub_base = ToString(Digest('\MathHub{'}));
560   my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);
561   return Invocation(T_CS('\includeassignment'), $keyval, T_OTHER($finalpath)); }##
562 DefKeyVal('inclprob','mhrepos','Semiverbatim');
563 DefMacro('\includemhassignment OptionalKeyVals:inclprob {'', \&includemhassignment);
564 \</hwexam.ltxml>

```

`\inputmhassignment` analogous

```

565 \<hwexam>
566 \newcommand\inputmhassignment[2][\metasetkeys{inclassig}{#1}%
567 \edef\mh@@repos{\mh@currentrepos}%
568 \ifx\inclassig@mhrepos\empty\else\mhcurrentrepos\inclassig@mhrepos\fi%
569 \inputassignment[#1]{\MathHub{\mh@currentrepos/source/#2}}%
570 \mhcurrentrepos\mh@@repos\clear@inclassig@keys}

```

```

571 </hwexam>
572 <*hwexam.ltxml>
573 sub inputmhassignment {
574   my ($gullet,$keyval,$arg2) = @_;
575   my $repo_path;
576   if ($keyval) {
577     $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
578   if (! $repo_path) {
579     $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }
580   else {
581     $keyval->setValue('mhrepos',undef); }
582   my $mathhub_base = ToString(Digest('\MathHub{ }'));
583   my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);
584   return Invocation(T_CS('\inputassignment'), $keyval, T_OTHER($finalpath)); }#$
585 DefMacro('\inputmhassignment OptionalKeyVals:inclprob {}', \&inputmhassignment);
586 </hwexam.ltxml>

```

## 4.10 Finale

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

```

587 <ltxml | modules.ltxml | structview.ltxml | omtex.ltxml | statements.ltxml | smultiling.ltxml | mikosides.ltxml | problem.

```