

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

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

November 8, 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>2</b>
<b>2</b>	<b>The User Interface</b>	<b>2</b>
2.1	Package Options . . . . .	2
2.2	<code>modules-mh</code> : MH Variants for Modules . . . . .	2
2.3	<code>omtext-mh</code> : MH Variants for OMText . . . . .	3
2.4	<code>smultiling-mh</code> : MH Variants for Multilinguality . . . . .	3
2.5	<code>structview-mh</code> : MH Variants for Structures and Views . . . . .	3
2.6	<code>mikoslides-mh</code> : Support for MiKo Slides . . . . .	3
2.7	<code>problem-mh</code> : Support for Problems . . . . .	4
2.8	<code>hwexam-mh</code> : Support for Assignments . . . . .	4
<b>3</b>	<b>Limitations</b>	<b>4</b>
<b>4</b>	<b>Implementation</b>	<b>5</b>
4.1	General Infrastructure . . . . .	5
4.2	<code>modules-mh</code> : MH Variants for Modules . . . . .	6
4.3	<code>omtext-mh</code> : MH Variants for OMText . . . . .	9
4.4	<code>smultiling-mh</code> : MH Variants for Multilinguality . . . . .	10
4.5	<code>structview-mh</code> : MH Variants for Structures and Views . . . . .	12

---

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

4.6	mikoslides-mh: Support for MiKo Slides . . . . .	15
4.7	problem-mh: Support for Problems . . . . .	15
4.8	hwexam-mh: Support for Assignments . . . . .	16
4.9	Finale . . . . .	18

# 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 be 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 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 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 smultiling-mh: MH Variants for Multilinguality

1 2

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

3

## 2.6 mikosides-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}}
```

---

<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

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

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

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

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

```
\mhframeimage{baz/foobar}
```

## 2.7 problem-mh: Support for Problems

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

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

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

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

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

## 2.8 hwexam-mh: Support for Assignments

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

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

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

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

## 3 Limitations

In this section we document known limitations. If you want to help alleviate them, please feel free to contact the package author. Some of them are currently discussed in the `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 `\endpackage`) and the  $\text{\LaTeX}$ 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  $\text{\LaTeX}$ ML binding files and the base package.

```

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

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

```

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

```

15 \package
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 \package
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 exist}%
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@crepos{\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@crepos}% 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 \let\mhinput\mhinputref%

```

### 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.

```

163 <*omtext>
164 \ProvidesPackage{omtext-mh}[2015/11/04 v1.0 MathHub support for the sTeX omtex package]
165 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{omtext}}
166 \ProcessOptions
167 \RequirePackage{omtext}
168 \RequirePackage{mathhub}
169 </omtext>
170 <omtext.ltxml>
171 \DeclareOption(undef,sub{PassOptions('omtext','sty',ToString(Digest(T_CS('\CurrentOption')))); }
172 \ProcessOptions();
173 \RequirePackage('omtext');
174 \RequirePackage('mathhub');
175 </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`.

```

176 <*omtext>
177 \addmetakey{Gin}{mhrepos}
178 \newcommand\mhgraphics[2][]{\metasetkeys{Gin}{#1}%
179 \edef\mh@@repos{\mh@currentrepos}%
180 \ifx\Gin@mhrepos@empty\mygraphics[#1]{\MathHub{\mh@currentrepos/source/#2}}%
181 \else\mygraphics[#1]{\MathHub{\Gin@mhrepos/source/#2}}\fi
182 \def\Gin@mhrepos{}\mhcurrentrepos\mh@@repos}
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 </omtext>
187 <omtext.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 smultiling-mh: MH Variants for Multilinguality

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

```

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

```

`mhmodnl:`

```

218 <*smultiling>
219 \addmetakey{mhmodnl}{repos}
220 \addmetakey{mhmodnl}{path}
221 \addmetakey*{mhmodnl}{title}
222 \addmetakey*{mhmodnl}{creators}
223 \addmetakey*{mhmodnl}{contributors}
224 \addmetakey{mhmodnl}{srccite}
225 \addmetakey{primary}{mhmodnl}[yes]
226 </smultiling>
227 <*smultiling.ltxml>
228 DefKeyVal('mhmodnl', 'title', 'Semiverbatim');
229 DefKeyVal('mhmodnl', 'repos', 'Semiverbatim');
230 DefKeyVal('mhmodnl', 'path', 'Semiverbatim');
231 DefKeyVal('mhmodnl', 'creators', 'Semiverbatim');
232 DefKeyVal('mhmodnl', 'contributors', 'Semiverbatim');
233 DefKeyVal('mhmodnl', 'primary', 'Semiverbatim');
234 </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.

```

235 <*smultiling>
236 \newenvironment{mhmodnl}[3][[]]{\metasetkeys{mhmodnl}{#1}%
237 \def\@test{#1}\ifx\@test\@empty\begin{module}[id=#2.#3]\else\begin{module}[id=#2.#3,#1]\fi%

```

```

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

```

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

```

278 \<smultiling.ltxml>RawTeX('
279 \<smultiling | smultiling.ltxml>
280 \newenvironment{mhviewsig}[4][[]{\def\@test{#1}\ifx\@test\empty%
281 \begin{mhview}[id=#2,ext=tex]{#3}{#4}\else%
282 \begin{mhview}[id=#2,#1,ext=tex]{#3}{#4}\fi}
283 {\end{mhview}}

```

EdN:4

`mhviewnl` The `mhviewnl` environment is just a layer over the `mhviewsketch` environment with the keys and language suitably adapted.<sup>4</sup>

```
284 \newenvironment{mhviewnl}[5][]{\def\@test{#1}\ifx\@test\@empty%
285 \begin{mhviewsketch}[id=#2.#5,ext=tex]{#3}{#4}\else%
286 \begin{mhviewsketch}[id=#2.#5,#1,ext=tex]{#3}{#4}\fi}
287 {\end{mhviewsketch}}
288 \</smultiling | smultiling.ltxml>
289 \<smultiling.ltxml>');
```

## 4.5 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.

```
290 \<*structview>
291 \ProvidesPackage{structview-mh}[2015/11/04 v1.0 MathHub support for the sTeX structview package]
292 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{structview}}
293 \ProcessOptions
294 \RequirePackage{structview}
295 \RequirePackage{mathhub}
296 \</structview>
297 \<*structview.ltxml>
298 \DeclareOption{undef,sub{PassOptions('structview','sty',ToString(Digest(T_CS('\CurrentOption'))))}}
299 \ProcessOptions();
300 \RequirePackage('structview');
301 \RequirePackage('mathhub');
302 \</structview.ltxml>
```

`importmhmodulevia`

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

313 \<*structview>
314 \srefaddidkey{mhview}
315 \addmetakey{mhview}{display}
316 \addmetakey{mhview}{creators}
317 \addmetakey{mhview}{contributors}
318 \addmetakey{mhview}{srccite}
319 \addmetakey*{mhview}{title}
```

---

<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)`!

```

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

```

mhview the MathHub version

```

339 \<structview>
340 \newenvironment{mhview}[3][{}]{% keys, from, to
341   \metasetkeys{mhview}{#1}%
342   \sref@target%
343   \begin{@mhview}{#2}{#3}%
344   \view@heading{#2}{#3}{\mhview@display}{\mhview@title}%
345 }{%
346   \end{@mhview}%
347   \ignorespaces%
348 }%
349 \ifmod@show\surroundwithmdframed{mhview}\fi
350 \</structview>
351 \<structview.ltxml>
352 DefMacroI(T_CS('\begin{mhview}'), 'OptionalKeyVals: mhview {}{}', sub {
353   my ($gullet, $keyvals, $from_arg, $to_arg) = @_;
354   my $from = ToString(Digest($from_arg));
355   my $to = ToString(Digest($to_arg));
356   AssignValue(from_module => $from);
357   AssignValue(to_module => $to);
358   my $from_repos = ToString(GetKeyVal($keyvals, 'fromrepos'));
359   my $to_repos = ToString(GetKeyVal($keyvals, 'torepos'));
360   my $repos = LookupValue('current_repos');
361   my $from_path = ToString(GetKeyVal($keyvals, 'frompath'));
362   my $to_path = ToString(GetKeyVal($keyvals, 'topath'));
363   my $ext = ToString(GetKeyVal($keyvals, 'ext')) if $keyvals;
364   $ext = 'sms' unless $ext;
365   my $current_repos = LookupValue('current_repos');
366   if (!$from_repos) { $from_repos = $current_repos; }
367   if (!$to_repos) { $to_repos = $current_repos; }

```

```

368 return (
369     Tokenize("\importMHmoduleI[repos=$from_repos,path=$from_path,ext=$ext]{$from}")->unlist,
370     Tokenize("\importMHmoduleI[repos=$to_repos,path=$to_path,ext=$ext]{$to}")->unlist,
371     Invocation(T_CS('\begin{viewenv}'),$keyvals,$from_arg,$to_arg)->unlist
372 );
373 });
374 DefMacroI('\end{mhview}',undef,'\end{viewenv}');
375 \</structview.ltxml>

```

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

```

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

```

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

```

382 <*structview>
383 \newenvironment{mhviewsketch}[3][{}]{%
384     \metasetkeys{mhview}{#1}%
385     \sref@target%
386     \begin{@mhview}{#2}{#3}%
387     \view@heading{#2}{#3}{\mhview@display}{\mhview@title}%
388 }{}%
389     \end{@mhview}%
390     \ignorespaces%
391 }%
392 \ifmod@show\surroundwithmdframed{mhviewsketch}\fi
393 \</structview>
394 <*structview.ltxml>
395 DefMacroI(T_CS('\begin{mhviewsketch}'),'OptionalKeyVals:mhview {}{}', sub {
396     my ($gullet, $keyvals, $from_arg, $to_arg) = @_;
397     my $from = ToString(Digest($from_arg));
398     my $to = ToString(Digest($to_arg));
399     my $from_repos = ToString(GetKeyVal($keyvals,'fromrepos'));
400     my $to_repos = ToString(GetKeyVal($keyvals,'torepos'));
401     my $repos = LookupValue('current_repos');
402     my $from_path = ToString(GetKeyVal($keyvals,'frompath'));
403     my $to_path = ToString(GetKeyVal($keyvals,'topath'));
404     my $ext = ToString(GetKeyVal($keyvals,'ext')) if $keyvals;
405     $ext = 'sms' unless $ext;
406     my $current_repos = LookupValue('current_repos');
407     if (!$from_repos) { $from_repos = $current_repos; }
408     if (!$to_repos) { $to_repos = $current_repos; }
409     return (
410         Tokenize("\importMHmoduleI[repos=$from_repos,path=$from_path,ext=$ext]{$from}")->unlist,
411         Tokenize("\importMHmoduleI[repos=$to_repos,path=$to_path,ext=$ext]{$to}")->unlist,
412         Invocation(T_CS('\begin{viewsketchenv}'),$keyvals,$from_arg,$to_arg)->unlist
413     );

```

```

414 });
415 DefMacroI('\end{mhviewsketch}',undef,'\end{viewsketchenv}');
416 </structview.ltxml>

```

## 4.6 mikoslides-mh: Support for MiKo Slides

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

```

417 <*mikoslides>
418 \ProvidesPackage{mikoslides-mh}[2015/11/04 v1.0 MathHub support for the sTeX mikoslides package]
419 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{mikoslides}}
420 \ProcessOptions
421 \RequirePackage{mikoslides}
422 \RequirePackage{mathhub}
423 </mikoslides>
424 <*mikoslides.ltxml>
425 DeclareOption(undef,sub{PassOptions('mikoslides','sty',ToString(Digest(T_CS('\CurrentOption'))))}
426 ProcessOptions();
427 RequirePackage('mikoslides');
428 RequirePackage('mathhub');
429 </mikoslides.ltxml>

```

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

```

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

```

## 4.7 problem-mh: Support for Problems

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

```

445 <*problem>
446 \ProvidesPackage{problem-mh}[2015/11/04 v1.0 MathHub support for the sTeX problem package]
447 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{problem}}
448 \ProcessOptions

```



```

449 \RequirePackage{problem}
450 \RequirePackage{mathhub}
451 \end{problem}
452 \begin{problem.ltxml}
453 \DeclareOption{undef}{\sub{PassOptions('problem','sty',ToString(Digest(T_CS('CurrentOption'))));}
454 \ProcessOptions();
455 \RequirePackage('problem');
456 \RequirePackage('mathhub');
457 \end{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`.

```

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

```

## 4.8 hwexam-mh: Support for Assignments

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

```

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

```

```

488 <hwexam.ltxml>
489 DeclareOption(undef,sub{PassOptions('hwexam','sty',ToString(Digest(T_CS('\CurrentOption')))); }
490 ProcessOptions();
491 RequirePackage('hwexam');
492 RequirePackage('mathhub');
493 </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`.

```

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

```

`\inputmhassignment` analogous

```

517 <*package>
518 \newcommand\inputmhassignment[2][\metasetkeys{inclassig}{#1}%
519 \edef\mh@@repos{\mh@currentrepos}%
520 \ifx\inclassig@mhrepos\empty\else\mhcurrentrepos\inclassig@mhrepos\fi%
521 \inputassignment[#1]{\MathHub{\mh@currentrepos/source/#2}}%
522 \mhcurrentrepos\mh@@repos\clear@inclassig@keys}
523 </package>
524 <*ltxml>
525 sub inputmhassignment {
526   my ($gullet,$keyval,$arg2) = @_ ;
527   my $repo_path;
528   if ($keyval) {
529     $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }

```

```

530 if (! $repo_path) {
531   $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }
532 else {
533   $keyval->setValue('mhrepos',undef); }
534 my $mathhub_base = ToString(Digest('\MathHub{ }'));
535 my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);
536 return Invocation(T_CS('\inputassignment'), $keyval, T_OTHER($finalpath)); }#$
537 DefMacro('\inputmhassignment OptionalKeyVals:inclprob { }', \&inputmhassignment);
538 </ltxml>

```

## 4.9 Finale

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

```

539 <*ltxml>
540 1;
541 </ltxml>

```