

# 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>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>smultiling-mh</code> : MH Variants for Multilinguality . . . . .	4
2.5	<code>structview-mh</code> : MH Variants for Structures and Views . . . . .	4
2.6	<code>mikoslides-mh</code> : Support for MiKo Slides . . . . .	4
2.7	<code>problem-mh</code> : Support for Problems . . . . .	5
2.8	<code>hwexam-mh</code> : Support for Assignments . . . . .	5
<b>3</b>	<b>Limitations</b>	<b>5</b>
<b>4</b>	<b>Implementation</b>	<b>6</b>
4.1	General Infrastructure . . . . .	6
4.2	<code>modules-mh</code> : MH Variants for Modules . . . . .	7
4.3	<code>omtext-mh</code> : MH Variants for OMText . . . . .	10
4.4	<code>smultiling-mh</code> : MH Variants for Multilinguality . . . . .	11
4.5	<code>structview-mh</code> : MH Variants for Structures and Views . . . . .	13

---

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

4.6	mikoslides-mh: Support for MiKo Slides . . . . .	16
4.7	problem-mh: Support for Problems . . . . .	17
4.8	hwexam-mh: Support for Assignments . . . . .	18
4.9	Finale . . . . .	19

# 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 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 smultiling-mh: MH Variants for Multilinguality

1 2

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

3

## 2.6 mikoslides-mh: Support for MiKo Slides

`\mhframeimage` 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](https://github.com/sTeX/sTeX)].

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 `\ltxml` 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 \ltxml | modules.ltxml | omtex.ltxml | smultiling.ltxml | mikosides.ltxml | problem.ltxml | hwexam.ltxml
2 \package \LaTeXML::Package::Pool;
3 \use strict;
4 \use \LaTeXML::Package;
5 \ltxml | modules.ltxml | omtex.ltxml | smultiling.ltxml | mikosides.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 \ltxml
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 \ltxml
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 <*txml>
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 </txml>#&

```

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

```

39 <*package>
40 \def\modules@@first#1/#2;{#1}
41 \newcommand\libinput[1]{\def\@libfile{\MathHub{\mh@currentrepos/lib/#1}}%
42 \IfFileExists{\@libfile}{\input\@libfile}%
43 {\edef\@group{\expandafter\modules@@first\mh@currentrepos;}
44 \edef\@inffile{\MathHub{\@group/meta-inf/lib/#1}}
45 \IfFileExists{\@inffile}{\input{\@inffile}}%
46 {\PackageError{modules}
47   {Library file missing, cannot input #1\MessageBreak%
48     Both \@libfile.tex\MessageBreak and \@inffile.tex\MessageBreak do not exist}%
49   {Check whether the file name is correct}}}%
50 </package>
51 <*txml>
52 DefMacro('\'modules@@first#1/#2;\'',\'\'#1\'');
53 DefMacro('\'libinput{\'',\'\'def\@libfile{\MathHub{\mh@currentrepos/lib/#1}\'');
54 </txml>

```

## 4.2 modules-mh: MH Variants for Modules

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

```

55 <*modules>
56 \ProvidesPackage{modules-mh}[2015/11/04 v1.0 MathHub support for the sTeX modules package]
57 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{modules}}
58 \ProcessOptions
59 \RequirePackage{modules}
60 \RequirePackage{mathhub}
61 </modules>
62 <*modules.ltxml>
63 DeclareOption(undef,sub{PassOptions('modules','sty',ToString(Digest(T_CS('CurrentOption'))));
64 ProcessOptions();
65 RequirePackage('modules');
66 RequirePackage('mathhub');
67 </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 comparison with an \expandafter, since the values may be passed on from other key bindings. Parameters will be passed to \importmodule.

```

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

```



```

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

```

and now the analogs

\usemhmodule

```

127 <*modules>
128 \newcommand\usemhmodule[2] [] {%
129 \metasetkeys{importmhmodule}{#1}%
130 \ifx\importmhmodule@path\empty%
131 \usemodule[ext=\importmhmodule@ext,id=\importmhmodule@id]{#2}%
132 \else%
133 \edef\mh@@repos{\mh@currentrepos}%
134 \ifx\importmhmodule@repos\empty%
135 \else%
136 \mhcurrentrepos{\importmhmodule@repos}%
137 \fi%
138 \usemodule[load=MathHub{\mh@currentrepos/source/\importmhmodule@path},ext=\importmhmodule@
139 \mhcurrentrepos\mh@@repos%
140 \fi%
141 \ignorespaces%
142 }%
143 </modules>
144 <*modules.ltxml>
145 DefConstructor('\usemhmodule OptionalKeyVals:importmhmodule {}',
146 " <omdoc:uses from='?&GetKeyVal(#1,'load')(&canonical_omdoc_path(&GetKeyVal(#1,'load')))(###
147 afterDigest => \&importMHmoduleI);
148 </modules.ltxml>

```

\mhinputref

```

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

```

```

158 \inputref{\MathHub{\mh@currentrepos/source/#2}}%
159 \mhcurrentrepos\mh@repos%
160 \ignorespaces%
161 }%
162 </modules | modules.ltxml>
163 <modules.ltxml>' );

```

`\mhinput`

```

164 \let\mhinput\mhinputref%

```

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

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

```

165 <*omtext>
166 \ProvidesPackage{omtext-mh}[2015/11/04 v1.0 MathHub support for the sTeX omtex package]
167 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{omtext}}
168 \ProcessOptions
169 \RequirePackage{omtext}
170 \RequirePackage{mathhub}
171 </omtext>
172 <*omtext.ltxml>
173 \DeclareOption(undef,sub{PassOptions('omtext','sty',ToString(Digest(T_CS('CurrentOption')))); }
174 \ProcessOptions();
175 \RequirePackage('omtext');
176 \RequirePackage('mathhub');
177 </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`.

```

178 <*omtext>
179 \addmetakey{Gin}{mhrepos}
180 \newcommand\mhgraphics[2] [] {\metasetkeys{Gin}{#1}}%
181 \edef\mh@repos{\mh@currentrepos}%
182 \ifx\Gin@mhrepos\empty\mygraphics[#1]{\MathHub{\mh@currentrepos/source/#2}}%
183 \else\mygraphics[#1]{\MathHub{\Gin@mhrepos/source/#2}}\fi
184 \def\Gin@mhrepos{\mhcurrentrepos\mh@repos}
185 \newcommand\mhgraphics[2] [] {\begin{center}\mhgraphics[#1]{#2}\end{center}}
186 \newcommand\mhgraphics[2] [] {\fbox{\mhgraphics[#1]{#2}}}
187 \newcommand\mhcbgraphics[2] [] {\begin{center}\fbox{\mhgraphics[#1]{#2}}\end{center}}
188 </omtext>
189 <*omtext.ltxml>
190 sub mhgraphics {
191   my ($gullet,$keyval,$arg2) = @_;
192   my $repo_path;
193   if ($keyval) {
194     $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
195   if (! $repo_path) {
196     $repo_path = ToString(Digest(T_CS('mh@currentrepos'))); }

```

```

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

```

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

```

`mhmodnl:`

```

220 <*smultiling>
221 \addmetakey{mhmodnl}{repos}
222 \addmetakey{mhmodnl}{path}
223 \addmetakey*{mhmodnl}{title}
224 \addmetakey*{mhmodnl}{creators}
225 \addmetakey*{mhmodnl}{contributors}
226 \addmetakey{mhmodnl}{srccite}
227 \addmetakey{primary}{mhmodnl}[yes]
228 </smultiling>
229 <*smultiling.ltxml>
230 DefKeyVal('mhmodnl','title','Semiverbatim');
231 DefKeyVal('mhmodnl','repos','Semiverbatim');
232 DefKeyVal('mhmodnl','path','Semiverbatim');
233 DefKeyVal('mhmodnl','creators','Semiverbatim');
234 DefKeyVal('mhmodnl','contributors','Semiverbatim');
235 DefKeyVal('mhmodnl','primary','Semiverbatim');
236 </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.

```

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

```

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

```

280 <smultiling.ltxml>RawTeX('
281 <smultiling | smultiling.ltxml>

```

```

282 \newenvironment{mhviewsig}[4][]{\def\@test{#1}\ifx\@test\@empty%
283 \begin{mhview}[id=#2,ext=tex]{#3}{#4}\else%
284 \begin{mhview}[id=#2,#1,ext=tex]{#3}{#4}\fi}
285 {\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>

```

286 \newenvironment{mhviewnl}[5][]{\def\@test{#1}\ifx\@test\@empty%
287 \begin{mhviewsketch}[id=#2.#5,ext=tex]{#3}{#4}\else%
288 \begin{mhviewsketch}[id=#2.#5,#1,ext=tex]{#3}{#4}\fi}
289 {\end{mhviewsketch}}
290 \</smultiling | smultiling.ltxml>
291 \<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.

```

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

```

**importmhmodulevia**

```

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

315 \<structview>
316 \srefaddidkey{mhview}
317 \addmetakey{mhview}{display}

```

<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`!

```

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

```

mhview the MathHub version

```

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

```

```

366 $ext = 'sms' unless $ext;
367 my $current_repos = LookupValue('current_repos');
368 if (!$from_repos) { $from_repos = $current_repos; }
369 if (!$to_repos) { $to_repos = $current_repos; }
370 return (
371     Tokenize("\importMHmoduleI[repos=$from_repos,path=$from_path,ext=$ext]{$from}")->unlist,
372     Tokenize("\importMHmoduleI[repos=$to_repos,path=$to_path,ext=$ext]{$to}")->unlist,
373     Invocation(T_CS('\begin{viewenv}'), $keyvals, $from_arg, $to_arg)->unlist
374 );
375 });
376 DefMacroI('\end{mhview}', undef, '\end{viewenv}');
377 </structview.ltxml>

```

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

```

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

```

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

```

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

```

```

412 Tokenize("\importMHmoduleI[repos=$from_repos,path=$from_path,ext=$ext]{$from}")->unlist,
413 Tokenize("\importMHmoduleI[repos=$to_repos,path=$to_path,ext=$ext]{$to}")->unlist,
414 Invocation(T_CS('\begin{viewsketchenv}'),$keyvals,$from_arg,$to_arg)->unlist
415 );
416 });
417 DefMacroI('\end{mhviewsketch}',undef,'\end{viewsketchenv}');
418 </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.

```

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

```

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

```

432 <mikoslides>\addmetakey{Gin}{mhrepos}
433 <mikoslides.ltxml>DefKeyVal('Gin','mhrepos','Semiverbatim');
434 <mikoslides.ltxml>RawTeX('
435 <*mikoslides.ltxml | mikoslides>
436 \newcommand\mhframeimage[2][{}]{%
437   \metasetkeys{Gin}{#1}%
438   \edef\mh@@repos{\mh@currentrepos}%
439   \ifx\Gin@mhrepos\empty%
440     \frameimage[#1]{\MathHub{\mh@currentrepos/source/#2}}%
441   \else%
442     \frameimage[#1]{\MathHub{\Gin@mhrepos/source/#2}}%
443   \fi%
444 }%
445 </mikoslides.ltxml | mikoslides>
446 <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.



```

447 <*problem>
448 \ProvidesPackage{problem-mh}[2015/11/04 v1.0 MathHub support for the sTeX problem package]
449 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{problem}}
450 \ProcessOptions
451 \RequirePackage{problem}
452 \RequirePackage{mathhub}
453 </problem>
454 <*problem.ltxml>
455 DeclareOption(undef,sub{PassOptions('problem','sty',ToString(Digest(T_CS('\CurrentOption'))));
456 ProcessOptions();
457 RequirePackage('problem');
458 RequirePackage('mathhub');
459 </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`.

```

460 <*problem>
461 \newcommand\includemhproblem[2][\metasetkeys{inclprob}{#1}%
462 \edef\mh@@repos{\mh@currentrepos}%
463 \ifx\inclprob\mhrepos\empty\else\mhcurrentrepos\inclprob\mhrepos\fi%
464 \input{\MathHub{\mh@currentrepos/source/#2}}%
465 \mhcurrentrepos\mh@@repos\clear@inclprob@keys}
466 </problem>
467 <*problem.ltxml>
468 sub includemhproblem {
469   my ($gullet,$keyval,$arg2) = @_ ;
470   my $repo_path;
471   if ($keyval) {
472     $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
473   if (! $repo_path) {
474     $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }
475   else {
476     $keyval->setValue('mhrepos',undef); }
477   my $mathhub_base = ToString(Digest('\MathHub{'}));
478   my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);
479   return Invocation(T_CS('\includeproblem'), $keyval, T_OTHER($finalpath)); }##
480 DefKeyVal('inclprob','mhrepos','Semiverbatim');
481 DefMacro('\includemhproblem OptionalKeyVals:inclprob {}', \&includemhproblem);
482 </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.

```

483 <*hwexam>
484 \ProvidesPackage{hwexam-mh}[2015/11/04 v1.0 MathHub support for the sTeX hwexam package]
485 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{hwexam}}

```

```

486 \ProcessOptions
487 \RequirePackage{hwexam}
488 \RequirePackage{mathhub}
489 \</hwexam>
490 \<hwexam.ltxml>
491 DeclareOption(undef,sub{PassOptions('hwexam','sty',ToString(Digest(T_CS('\CurrentOption')))); }
492 ProcessOptions();
493 RequirePackage('hwexam');
494 RequirePackage('mathhub');
495 \</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`.

```

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

```

`\inputmhassignment` analogous

```

519 \<package>
520 \newcommand\inputmhassignment[2][\metasetkeys{inclassig}{#1}%
521 \edef\mh@@repos{\mh@currentrepos}%
522 \ifx\inclassig@mhrepos\@empty\else\mhcurrentrepos\inclassig@mhrepos\fi%
523 \inputassignment[#1]{\MathHub{\mh@currentrepos/source/#2}}%
524 \mhcurrentrepos\mh@@repos\clear@inclassig@keys}
525 \</package>
526 \<ltxml>
527 sub inputmhassignment {

```

```

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

```

## 4.9 Finale

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

```

541 <*ltxml>
542 1;
543 </ltxml>

```