

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

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

November 17, 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 . . . . .	12
4.6	<code>structview-mh</code> : MH Variants for Structures and Views . . . . .	14
4.7	<code>mikoslides-mh</code> : Support for MiKo Slides . . . . .	17
4.8	<code>problem-mh</code> : Support for Problems . . . . .	17
4.9	<code>hwexam-mh</code> : Support for Assignments . . . . .	18
4.10	<code>tikzinput-mh</code> : Support for Assignments . . . . .	19
4.11	Finale . . . . .	20

# 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 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 `\endpackage`) and the  $\text{\LaTeX}$ XML 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  $\text{\LaTeX}$ XML binding files in the base package.

```

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

```

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

```

9 \package
10 \RequirePackage{keyval}
11 \endpackage
12 \beginltxml
13 \RequirePackage('keyval');
14 \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.

```

15 \package
16 \newcommand\mhcurrentrepos[1]{%
17   \edef\@test{#1}%
18   \ifx\@test\mh@currentrepos% if new dir = old dir
19     \relax% no need to change
20   \else%
21     \protected@write\@auxout{}{\string\@mhcurrentrepos{#1}}%
22   \fi%
23   \@mhcurrentrepos{#1}% define mh@currentrepos
24 }%
25 \newcommand\@mhcurrentrepos[1]{\edef\mh@currentrepos{#1}}%
26 \endpackage
27 \beginltxml
28 \DefMacro(' \mhcurrentrepos{ }', '\@mhcurrentrepos{#1}');
29 \DefMacro(' \@mhcurrentrepos{ }', '\def\mh@currentrepos{#1}\@mhcurrentrepos{#1}');
30 \DefConstructor(' \@mhcurrentrepos{ }', '',
31   afterDigest => sub{ AssignValue('current_repos', ToString($_[1]->getArg(1)), 'global'); } );
32 \endltxml#\$

```

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

```

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

```

58 <*modules>
59 \ProvidesPackage{modules-mh}[2015/11/04 v1.0 MathHub support for the sTeX modules package]
60 \RequirePackage{mathhub}
61 </modules>
62 <*modules.ltxml>
63 RequirePackage('mathhub');
64 </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`.

```

65 <*modules>
66 \srefaddidkey{importmhmodule}%

```



```

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

```

```

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

```

and now the analogs

\usemhmodule

```

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

```

\mhinputref

```

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

```

\mhinput

```
161 <*modules>
162 \let\mhinput\mhinputref%
163 </modules>
```

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

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

```
164 <*omtext>
165 \ProvidesPackage{omtext-mh}[2015/11/04 v1.0 MathHub support for the sTeX omtex package]
166 \RequirePackage{mathhub}
167 </omtext>
168 <*omtext.ltxml>
169 \RequirePackage('mathhub');
170 </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.

```
171 <*omtext>
172 \define@key{Gin}{mhrepos}{\csxdef\Gin@mhrepos{#1}}
173 \newcommand\mhgraphics[2][]{\setkeys{Gin}{#1}%
174 \edef\mh@currentrepos{\mh@currentrepos}%
175 \ifx\Gin@mhrepos\empty\mygraphics[#1]{\MathHub{\mh@currentrepos/source/#2}}}%
176 \else\mygraphics[#1]{\MathHub{\Gin@mhrepos/source/#2}}\fi
177 \def\Gin@mhrepos{\mhcurrentrepos\mh@currentrepos}
178 \newcommand\mhgraphics[2][]{\begin{center}\mhgraphics[#1]{#2}\end{center}}
179 \newcommand\mhgraphics[2][]{\fbox{\mhgraphics[#1]{#2}}}
180 \newcommand\mhcbgraphics[2][]{\begin{center}\fbox{\mhgraphics[#1]{#2}}\end{center}}
181 </omtext>
182 <*omtext.ltxml>
183 sub mhgraphics {
184   my ($gullet,$keyval,$arg2) = @_ ;
185   my $repo_path;
186   if ($keyval) {
187     $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
188   if (! $repo_path) {
189     $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }
190   else {
191     $keyval->setValue('mhrepos',undef); }
192   my $mathhub_base = ToString(Digest('\MathHub{'}));
193   my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);
194   return Invocation(T_CS('@includegraphicx'), $keyval, T_OTHER($finalpath)); }#$
195 DefKeyVal('Gin','mhrepos','Semiverbatim');
196 DefMacro('\mhgraphics OptionalKeyVals:Gin {}', \&mhgraphics);
197 DefMacro('\mhgraphics []{}', '\begin{center}\mhgraphics[#1]{#2}\end{center}');
198 DefMacro('\mhgraphics []{}', '\fbox{\mhgraphics[#1]{#2}}');
199 </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.

```
200 <*statements>
201 \ProvidesPackage{statements-mh}[2015/11/04 v1.0 MathHub support for the sTeX statements package]
202 \RequirePackage{mathhub}
203 </statements>
204 <statements.ltxml>
205 \RequirePackage('mathhub');
206 </statements.ltxml>

207 <*statements>
208 \let\usemhvocab=\usemhmodule
209 </statements>
210 <statements.ltxml>
211 \DefMacro('usemhvocab','usemhmodule');
212 </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.

```
213 <*smultiling>
214 \ProvidesPackage{smultiling-mh}[2015/11/04 v1.0 MathHub support for the sTeX smultiling package]
215 \RequirePackage{mathhub}
216 </smultiling>
217 <smultiling.ltxml>
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]\else\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>');

```

## 4.6 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.

```

292 <*structview>
293 \ProvidesPackage{structview-mh}[2015/11/04 v1.0 MathHub support for the sTeX structview package]
294 \RequirePackage{mathhub}
295 </structview>
296 <*structview.ltxml>
297 \RequirePackage('mathhub');
298 </structview.ltxml>

```

**importmhmodulevia**

```

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

309 <*structview>
310 \srefaddidkey{mhview}
311 \addmetakey{mhview}{display}
312 \addmetakey{mhview}{creators}
313 \addmetakey{mhview}{contributors}
314 \addmetakey{mhview}{src cite}
315 \addmetakey*{mhview}{title}
316 \addmetakey{mhview}{fromrepos}

```

<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)!`

```

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

```

mhview the MathHub version

```

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

```

```

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

```

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

```

372 <*structview>
373 \newenvironment{@mhview}[2]{%from, to
374 \importmhmodule[repos=\mhview@fromrepos,path=\mhview@frompath,ext=\mhview@ext]{#1}%
375 \importmhmodule[repos=\mhview@torepos,path=\mhview@topath,ext=\mhview@ext]{#2}%
376 }{}%
377 </structview>

```

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

```

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

```



```

411 DefMacroI('\end{mhviewsketch}',undef,'\end{viewsketchenv}');
412 \end{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.

```

413 \begin{mhviewsketch}
414 \ProvidesPackage{mikoslides-mh}[2015/11/04 v1.0 MathHub support for the sTeX mikoslides package]
415 \RequirePackage{mathhub}
416 \end{mhviewsketch}
417 \begin{mhviewsketch}
418 \RequirePackage('mathhub');
419 \end{mhviewsketch}

```

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

```

420 \def\mh@currentrepos{\csxdef\Gin@mhrepos{#1}}
421 \def\mh@mhrepos{\csxdef\Gin@mhrepos{#1}}
422 \def\mh@mhrepos{\csxdef\Gin@mhrepos{#1}}
423 \def\mh@mhrepos{\csxdef\Gin@mhrepos{#1}}
424 \newcommand\mhframeimage[2][{}]{%
425   \setkeys{Gin}{#1}%
426   \edef\mh@currentrepos{\mh@currentrepos}%
427   \ifx\Gin@mhrepos\empty%
428     \frameimage[1]{\MathHub{\mh@currentrepos/source/#2}}%
429   \else%
430     \frameimage[1]{\MathHub{\Gin@mhrepos/source/#2}}%
431   \fi%
432 }%
433 \end{mhviewsketch}
434 \end{mhviewsketch}

```

## 4.8 problem-mh: Support for Problems

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

```

435 \begin{mhviewsketch}
436 \ProvidesPackage{problem-mh}[2015/11/04 v1.0 MathHub support for the sTeX problem package]
437 \RequirePackage{mathhub}
438 \end{mhviewsketch}
439 \begin{mhviewsketch}
440 \RequirePackage('mathhub');
441 \end{mhviewsketch}

```

`\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`.

```

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

```

465 <*hwexam>
466 \ProvidesPackage{hwexam-mh}[2015/11/04 v1.0 MathHub support for the sTeX hwexam package]
467 \RequirePackage{mathhub}
468 </hwexam>
469 <*hwexam.ltxml>
470 RequirePackage('mathhub');
471 </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`.

```

472 <*hwexam>
473 \newcommand\includemhassignment[2][\metasetkeys{inclassig}{#1}%
474 \edef\mh@@repos{\mh@currentrepos}%
475 \ifx\inclassig@mhrepos\empty\else\mhcurrentrepos\inclassig@mhrepos\fi%
476 \includeassignment[#1]{\MathHub{\mh@currentrepos/source/#2}}%
477 \mhcurrentrepos\mh@@repos\clear@inclassig@keys}

```

```

478 </hwexam>
479 <*hwexam.ltxml>
480 sub includemhassignment {
481   my ($gullet,$keyval,$arg2) = @_;
482   my $repo_path;
483   if ($keyval) {
484     $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
485   if (! $repo_path) {
486     $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }
487   else {
488     $keyval->setValue('mhrepos',undef); }
489   my $mathhub_base = ToString(Digest('\MathHub{'}));
490   my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);
491   return Invocation(T_CS('\includeassignment'), $keyval, T_OTHER($finalpath)); }#$
492 DefKeyVal('inclprob','mhrepos','Semiverbatim');
493 DefMacro('\includemhassignment OptionalKeyVals:inclprob {}', \&includemhassignment);
494 </hwexam.ltxml>

```

\inputmhassignment analogous

```

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

```

## 4.10 tikzinput-mh: Support for Assignments

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

```

517 <*tikzinput>
518 \ProvidesPackage{tikzinput-mh}[2015/11/04 v1.0 MathHub support for the sTeX tikzinput package]
519 \RequirePackage{mathhub}

```

```

520 </tikzinput>
521 <*tikzinput.ltxml>
522 \RequirePackage('mathhub');
523 </tikzinput.ltxml>

524 <tikzinput.ltxml>\RawTeX('
525 <*tikzinput | tikzinput.ltxml>
526 \define@key{Gin}{mhrepos}{\csxdef\Gin@mhrepos{#1}}
527 \newcommand\mhtikzinput[2][]{\def\Gin@mhrepos{}\setkeys{Gin}{#1}%
528 \edef\mh@@repos{\mh@currentrepos}%
529 \ifx\Gin@mhrepos\@empty\tikzinput[#1]{\MathHub{\mh@currentrepos/source/#2}}%
530 \else\tikzinput[#1]{\MathHub{\Gin@mhrepos/source/#2}}\fi
531 \def\Gin@mhrepos{\mhcurrentrepos\mh@@repos}
532 \newcommand\cmhtikzinput[2][]{\begin{center}\mhtikzinput[#1]{#2}\end{center}}
533 </tikzinput | tikzinput.ltxml>
534 <tikzinput.ltxml>');

```

## 4.11 Finale

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

```

535 <ltxml | modules.ltxml | structview.ltxml | omtex.ltxml | statements.ltxml | multiling.ltxml | mikosides.ltxml | problem

```