

MathHub Support for \LaTeX^*

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

November 9, 2015

Abstract

The `sref` package is part of the \LaTeX collection, a version of $\text{\TeX}/\text{\LaTeX}$ that allows to markup $\text{\TeX}/\text{\LaTeX}$ documents semantically without leaving the document format, essentially turning $\text{\TeX}/\text{\LaTeX}$ into a document format for mathematical knowledge management (MKM).

The `mathhub` packages extend \LaTeX with support for the MathHub.info portal

Contents

1	Introduction	3
2	The User Interface	3
2.1	Package Options	3
2.2	<code>modules-mh</code> : MH Variants for Modules	3
2.3	<code>omtext-mh</code> : MH Variants for OMText	4
2.4	<code>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
3	Limitations	5
4	Implementation	6
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	16
4.8	hwexam-mh: Support for Assignments	17
4.9	Finale	19

1 Introduction

Much of the \LaTeX content is hosted on **MathHub** (<http://MathHub.info>), a portal and archive for flexiformal mathematics. **MathHub** offers GIT repositories (public and private escrow) for mathematical documentation projects, online and offline authoring and document development infrastructure, and a rich, interactive reading interface. The `modules` package supports repository-sensitive operations on **MathHub**.

Note that **MathHub** has two-level repository names of the form $\langle group \rangle / \langle repo \rangle$, where $\langle group \rangle$ is a **MathHub**-unique repository group and $\langle repo \rangle$ a repository name that is $\langle group \rangle$ -unique. The file and directory structure of a repository is arbitrary – except that it starts with the directory `source` because they are Math Archives in the sense of [HorIacJuc:cscpnrr11]. But this structure can be hidden from the \LaTeX author with **MathHub**-enabled versions of the \LaTeX macros, which are defined in this package.

Caveat if you want to use the **MathHub** support macros (let’s call them **mh-variants**), then every time a module is imported or a document fragment is included from another repos, the mh-variant `\importmhmodule` must be used, so that the “current repository” is set accordingly. To be exact, we only need to use mh-variants, if the imported module or included document fragment use mh-variants.

2 The User Interface

2.1 Package Options

none so far

2.2 modules-mh: MH Variants for Modules

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

```
\defpath{MathHub}{/user/foo/lmh/MathHub}
\importmodule[load=\MathHub{fooMH/bar/source/baz/foobar}]{foobar}
```

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

```
\importmhmodule[repos=fooMH/bar,path=baz/foobar]{foobar}
```

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

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

`\importmhmodule[path=baz/foobar]{foobar}`

if no file needs to be loaded, `\importmhmodule` is the same as `\importmodule`.

`\mhcurrentrepos` Of course, neither \LaTeX nor \LaTeXML know about the repositories when they are called from a file system, so we can use the `\mhcurrentrepos` macro to tell them. But this is only needed to initialize the infrastructure in the driver file. In particular, we do not need to set it in each module, since the `\importmhmodule` macro sets the current repository automatically.

`\usemhmodule` The `\usemhmodule` is the analog to `\usemodule`.

`\mhinputref` For this, the `modules` package supplies the mh-variants `\mhinputref` and
`\mhinput` `\mhinput` of the `\inputref` macro introduced above and normal \LaTeX `\input` macro.

2.3 omtext-mh: MH Variants for OMText

`\mhgraphics` The `\mhgraphics` macro is a variant of `\mycgraphics` with repository support. Instead of writing

```
\defpath{MathHub}{/user/foo/lmh/MathHub}
\mycgraphics{\MathHub{fooMH/bar/source/baz/foobar}}
```

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

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

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

2.4 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}}
```

¹EDNOTE: needs to be documented

²EDNOTE: mhmodsig seems to be missing what happened?

³EDNOTE: 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 \LaTeX package (all the code between `\package` and `\endpackage`) and the \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 \LaTeX XML binding files in the base package.

```

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

8 \package
9 \DeclareOption*{}
10 \ProcessOptions
11 \endpackage
12 \beginltxml
13 use LaTeXXML::Util::Pathname;
14 \DeclareOption(undef,sub {});
15 \ProcessOptions();
16 \endltxml

```

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

```

17 \package
18 \RequirePackage{keyval}
19 \endpackage
20 \beginltxml
21 \RequirePackage('keyval');
22 \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.

```

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

```

```

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

```

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

```

41 \end{package}
42 \def\modules@@first#1/#2;{#1}
43 \newcommand\libinput[1]{\def\@libfile{\MathHub{\mh@currentrepos/lib/#1}}%
44 \IfFileExists{\@libfile}{\input\@libfile}%
45 {\edef\@group{\expandafter\modules@@first\mh@currentrepos;}
46 \edef\@inffile{\MathHub{\@group/meta-inf/lib/#1}}
47 \IfFileExists{\@inffile}{\input\@inffile}%
48 {\PackageError{modules}
49   {Library file missing, cannot input #1\MessageBreak%
50     Both \@libfile.tex\MessageBreak and \@inffile.tex\MessageBreak do not exist}%
51   {Check whether the file name is correct}}}%
52 \end{package}
53 \end{xml}
54 DefMacro('modules@@first#1/#2;', '\', '#1');
55 DefMacro('libinput {', sub{
56   my ($gullet, $name) = @_;
57   $name = ToString($name);
58   #Relative paths for recursive search
59   my $libpath = "../..../lib/";
60   my $inffile = "../..../META-INF/";
61   my $file = pathname_find($name, types => ['tex'], paths =>
62     $libpath);
63   $file = pathname_find($name, types=>['tex'], paths=>inffile) unless $file;
64   # Singal error if the file cannot be found
65   LaTeXML::Package::InputContent($file, noerror=>1);
66 });
67 \end{xml}

```

4.2 modules-mh: MH Variants for Modules

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

```

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

```

```

73 \RequirePackage{mathhub}
74 \</modules>
75 \<*modules.ltxml>
76 DeclareOption(undef,sub{PassOptions('modules','sty',ToString(Digest(T_CS('\CurrentOption'))));
77 ProcessOptions();
78 RequirePackage('modules');
79 RequirePackage('mathhub');
80 \</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.

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

```



```

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

```

and now the analogs

\usemhmodule

```

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

```

```

161 </modules.ltxml>

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

```

\mhinput

```

177 <*package>
178 \let\mhinput\mhinputref%
179 </package>

```

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.

```

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

```

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

```

193 <*omtex>
194 \addmetakey{Gin}{mhrepos}
195 \newcommand\mhgraphics[2] [] {\metasetkeys{Gin}{#1}%
196 \edef\mh@@repos{\mh@currentrepos}%
197 \ifx\Gin\mhrepos\@empty\mygraphics[#1]{\MathHub{\mh@currentrepos/source/#2}}%

```

```

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

```

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

```

`mhmodnl:`

```

235 \<smultiling>
236 \addmetakey{mhmodnl}{repos}
237 \addmetakey{mhmodnl}{path}
238 \addmetakey*{mhmodnl}{title}
239 \addmetakey*{mhmodnl}{creators}

```

```

240 \addmetakey*{mhmodnl}{contributors}
241 \addmetakey{mhmodnl}{srccite}
242 \addmetakey{primary}{mhmodnl}[yes]
243 \smultiling
244 \smultiling.ltxml
245 DefKeyVal('mhmodnl','title','Semiverbatim');
246 DefKeyVal('mhmodnl','repos','Semiverbatim');
247 DefKeyVal('mhmodnl','path','Semiverbatim');
248 DefKeyVal('mhmodnl','creators','Semiverbatim');
249 DefKeyVal('mhmodnl','contributors','Semiverbatim');
250 DefKeyVal('mhmodnl','primary','Semiverbatim');
251 \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.

```

252 \smultiling
253 \newenvironment{mhmodnl}[3][\metasetkeys{mhmodnl}{#1}%
254 \def\@test{#1}\ifx\@test\empty\begin{module}[id=#2.#3]\else\begin{module}[id=#2.#3,#1]\fi%
255 \edef\@repos{\ifx\mhmodnl@repos\empty\mh@currentrepos\else\mhmodnl@repos}
256 \if@langfiles\importmhmodule[repos=\@repos,load=#2,ext=tex]{#2}\else
257 \ifx\mhmodnl@load\empty\importmodule{#2}\else\importmodule[ext=tex,load=\mhmodnl@load]{#2}\fi%
258 \fi}
259 \end{module}}
260 \smultiling
261 \smultiling.ltxml
262 DefEnvironment('mhmodnl' OptionalKeyVals:mhmodnl {}{}',
263     "?#excluded()(<omdoc:theory xml:id='#2.#3' >"
264     . "?&defined(&GetKeyVal(#1,'creators'))(<dc:creator>&GetKeyVal(#1,'creators')</dc:cr
265     . "?&defined(&GetKeyVal(#1,'title'))(<dc:title>&GetKeyVal(#1,'title')</dc:title>())"
266     . "?&defined(&GetKeyVal(#1,'contributors'))(<dc:contributor>&GetKeyVal(#1,'contribut
267     . "<omdoc:imports from='?&GetKeyVal(#1,'load')(&canonical_omdoc_path(&GetKeyVal(#1,'
268     . "#body"
269     . "</omdoc:theory>)",
270     afterDigestBegin=>sub {
271     my ($stomach, $whatsit) = @_;
272     my $keyval = $whatsit->getArg(1);
273     my $signature = ToString($whatsit->getArg(2));
274     my $language = ToString($whatsit->getArg(3));
275     my $repos = ToString(GetKeyVal($keyval,'torepos'));
276     my $current_repos = LookupValue('current_repos');
277     if (!$repos) { $repos = $current_repos; }
278     my $defpaths = LookupValue('defpath');
279     my $load_path = ($$defpaths{MathHub}).$repos.'/source/'. $signature;
280
281     if ($keyval) {
282     # If we're not given load, AND the langfiles option is in effect,
283     # default to #2
284     if ((! $keyval->getValue('path')) && (LookupValue('smultiling_langfiles')) {
285     $keyval->setValue('load',$load_path); }
286     # Always load a TeX file

```

```

287     $keyval->setValue('ext','tex');
288     $keyval->setValue('id',"${signature}.${language}"); }
289     module_afterDigestBegin(@_);
290     importmoduleI(@_);
291     return; },
292     afterDigest=>sub {
293         module_afterDigest(@_); });
294 </smultiling.ltxml>%$

```

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

```

295 <smultiling.ltxml>RawTeX('
296 <*smultiling | smultiling.ltxml>
297 \newenvironment{mhviewsig}[4][\def\@test{#1}\ifx\@test\@empty%
298 \begin{mhview}[id=#2,ext=tex]{#3}{#4}\else%
299 \begin{mhview}[id=#2,#1,ext=tex]{#3}{#4}\fi}
300 {\end{mhview}}

```

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

```

301 \newenvironment{mhviewnl}[5][\def\@test{#1}\ifx\@test\@empty%
302 \begin{mhviewsketch}[id=#2.#5,ext=tex]{#3}{#4}\else%
303 \begin{mhviewsketch}[id=#2.#5,#1,ext=tex]{#3}{#4}\fi}
304 {\end{mhviewsketch}}
305 </smultiling | smultiling.ltxml>
306 <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.

```

307 <*structview>
308 \ProvidesPackage{structview-mh}[2015/11/04 v1.0 MathHub support for the sTeX structview package]
309 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{structview}}
310 \ProcessOptions
311 \RequirePackage{structview}
312 \RequirePackage{mathhub}
313 </structview>
314 <*structview.ltxml>
315 \DeclareOption(undef,sub{PassOptions('structview','sty',ToString(Digest(T_CS('\CurrentOption'))))}
316 \ProcessOptions();
317 \RequirePackage('structview');
318 \RequirePackage('mathhub');
319 </structview.ltxml>

```

importmhmodulevia

⁴EDNOTE: MK: we have to do something about the `if@langfiles` situation here. But this is non-trivial, since we do not know the current path, to which we could append `.\lang`!

```

320 <structview.ltxml>RawTeX(
321 <*structview | structview.ltxml>
322 \newenvironment{importmhmodulevia}[3][{}]{%
323   \gdef\@doit{\importmhmodule[#1]{#2}{#3}}%
324   \ifmod@show\par\noindent importing module #2 via \@doit\fi
325 }{%
326   \aftergroup\@doit\ifmod@show end import\fi%
327 }%
328 </structview | structview.ltxml>
329 <structview.ltxml>');

330 <*structview>
331 \srefaddidkey{mhview}
332 \addmetakey{mhview}{display}
333 \addmetakey{mhview}{creators}
334 \addmetakey{mhview}{contributors}
335 \addmetakey{mhview}{srccite}
336 \addmetakey*{mhview}{title}
337 \addmetakey{mhview}{fromrepos}
338 \addmetakey{mhview}{torepos}
339 \addmetakey{mhview}{frompath}
340 \addmetakey{mhview}{topath}
341 \addmetakey[sms]{mhview}{ext}
342 </structview>
343 <*structview.ltxml>
344 DefKeyVal('mhview','id','Semiverbatim');
345 DefKeyVal('mhview','display','Semiverbatim');
346 DefKeyVal('mhview','creators','Semiverbatim');
347 DefKeyVal('mhview','contributors','Semiverbatim');
348 DefKeyVal('mhview','srccite','Semiverbatim');
349 DefKeyVal('mhview','title','Semiverbatim');
350 DefKeyVal('mhview','fromrepos','Semiverbatim');
351 DefKeyVal('mhview','torepos','Semiverbatim');
352 DefKeyVal('mhview','frompath','Semiverbatim');
353 DefKeyVal('mhview','topath','Semiverbatim');
354 DefKeyVal('mhview','ext','Semiverbatim');
355 </structview.ltxml>

```

mhview the MathHub version

```

356 <*structview>
357 \newenvironment{mhview}[3][{}]{% keys, from, to
358   \metasetkeys{mhview}{#1}%
359   \sref@target%
360   \begin{@mhview}{#2}{#3}%
361   \view@heading{#2}{#3}{\mhview@display}{\mhview@title}%
362 }{%
363   \end{@mhview}%
364   \ignorespaces%
365 }%
366 \ifmod@show\surroundwithmdframed{mhview}\fi

```

```

367 </structview>
368 <*structview.ltxml>
369 DefMacroI(T_CS('\begin{mhview}'), 'OptionalKeyVals:mhview {}{}', sub {
370   my ($gullet, $keyvals, $from_arg, $to_arg) = @_;
371   my $from = ToString(Digest($from_arg));
372   my $to = ToString(Digest($to_arg));
373   AssignValue(from_module => $from);
374   AssignValue(to_module => $to);
375   my $from_repos = ToString(GetKeyVal($keyvals, 'fromrepos'));
376   my $to_repos = ToString(GetKeyVal($keyvals, 'torepos'));
377   my $repos = LookupValue('current_repos');
378   my $from_path = ToString(GetKeyVal($keyvals, 'frompath'));
379   my $to_path = ToString(GetKeyVal($keyvals, 'topath'));
380   my $ext = ToString(GetKeyVal($keyvals, 'ext')) if $keyvals;
381   $ext = 'sms' unless $ext;
382   my $current_repos = LookupValue('current_repos');
383   if (!$from_repos) { $from_repos = $current_repos; }
384   if (!$to_repos) { $to_repos = $current_repos; }
385   return (
386     Tokenize("\importMHmoduleI[repos=$from_repos,path=$from_path,ext=$ext]{$from}")->unlist,
387     Tokenize("\importMHmoduleI[repos=$to_repos,path=$to_path,ext=$ext]{$to}")->unlist,
388     Invocation(T_CS('\begin{viewenv}'), $keyvals, $from_arg, $to_arg)->unlist
389   );
390 });
391 DefMacroI('\end{mhview}', undef, '\end{viewenv}');
392 </structview.ltxml>

```

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

```

393 <*structview>
394 \newenvironment{@mhview}[2]{%from, to
395   \importmhmodule[repos=\mhview@fromrepos,path=\mhview@frompath,ext=\mhview@ext]{#1}%
396   \importmhmodule[repos=\mhview@torepos,path=\mhview@topath,ext=\mhview@ext]{#2}%
397 }{}%
398 </structview>

```

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

```

399 <*structview>
400 \newenvironment{mhviewsketch}[3][{}%
401   \metasetkeys{mhview}{#1}%
402   \sref@target%
403   \begin{@mhview}{#2}{#3}%
404   \view@heading{#2}{#3}{\mhview@display}{\mhview@title}%
405 ]{}%
406   \end{@mhview}%
407   \ignorespaces%
408 ]%
409 \ifmod@show\surroundwithmdframed{mhviewsketch}\fi
410 </structview>
411 <*structview.ltxml>
412 DefMacroI(T_CS('\begin{mhviewsketch}'), 'OptionalKeyVals:mhview {}{}', sub {

```

```

413 my ($gullet, $keyvals, $from_arg, $to_arg) = @_;
414 my $from = ToString(Digest($from_arg));
415 my $to = ToString(Digest($to_arg));
416 my $from_repos = ToString(GetKeyVal($keyvals,'fromrepos'));
417 my $to_repos = ToString(GetKeyVal($keyvals,'torepos'));
418 my $repos = LookupValue('current_repos');
419 my $from_path = ToString(GetKeyVal($keyvals,'frompath'));
420 my $to_path = ToString(GetKeyVal($keyvals,'topath'));
421 my $ext = ToString(GetKeyVal($keyvals,'ext')) if $keyvals;
422 $ext = 'sms' unless $ext;
423 my $current_repos = LookupValue('current_repos');
424 if (!$from_repos) { $from_repos = $current_repos; }
425 if (!$to_repos) { $to_repos = $current_repos; }
426 return (
427   Tokenize("\\importMHmoduleI[repos=$from_repos,path=$from_path,ext=$ext]{$from}")->unlist,
428   Tokenize("\\importMHmoduleI[repos=$to_repos,path=$to_path,ext=$ext]{$to}")->unlist,
429   Invocation(T_CS('\begin{viewsketchenv}'),$keyvals,$from_arg,$to_arg)->unlist
430 );
431 });
432 DefMacroI('\end{mhviewsketch}',undef,'\end{viewsketchenv}');
433 </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.

```

434 <*mikoslides>
435 \ProvidesPackage{mikoslides-mh}[2015/11/04 v1.0 MathHub support for the sTeX mikoslides package]
436 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{mikoslides}}
437 \ProcessOptions
438 \RequirePackage{mikoslides}
439 \RequirePackage{mathhub}
440 </mikoslides>
441 <*mikoslides.ltxml>
442 DeclareOption(undef,sub{PassOptions('mikoslides','sty',ToString(Digest(T_CS('\CurrentOption'))))}
443 ProcessOptions();
444 RequirePackage('mikoslides');
445 RequirePackage('mathhub');
446 </mikoslides.ltxml>

```

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

```

447 <mikoslides>\addmetakey{Gin}{mhrepos}
448 <mikoslides.ltxml>DefKeyVal('Gin','mhrepos','Semiverbatim');
449 <mikoslides.ltxml>RawTeX(
450 *mikoslides.ltxml | mikoslides)
451 \newcommand\mhframeimage[2][{}]{%
452   \metasetkeys{Gin}{#1}%
453   \edef\mh@@repos{\mh@currentrepos}%

```



```

454 \ifx\Gin@mhrepos\@empty%
455   \frameimage[#1]{\MathHub{\mh@currentrepos/source/#2}}%
456 \else%
457   \frameimage[#1]{\MathHub{\Gin@mhrepos/source/#2}}%
458 \fi%
459 }%
460 </mikoslides.ltxml | mikoslides>
461 <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.

```

462 <*problem>
463 \ProvidesPackage{problem-mh}[2015/11/04 v1.0 MathHub support for the sTeX problem package]
464 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{problem}}
465 \ProcessOptions
466 \RequirePackage{problem}
467 \RequirePackage{mathhub}
468 </problem>
469 <*problem.ltxml>
470 \DeclareOption(undef,sub{PassOptions('problem','sty',ToString(Digest(T_CS('\CurrentOption'))));}
471 \ProcessOptions();
472 \RequirePackage('problem');
473 \RequirePackage('mathhub');
474 </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`.

```

475 <*problem>
476 \newcommand\includemhproblem[2][\metasetkeys{inclprob}]{#1}%
477 \edef\mh@@repos{\mh@currentrepos}%
478 \ifx\inclprob@mhrepos\@empty\else\mh@currentrepos\inclprob@mhrepos\fi%
479 \input{\MathHub{\mh@currentrepos/source/#2}}%
480 \mh@currentrepos\mh@@repos\clear@inclprob@keys}
481 </problem>
482 <*problem.ltxml>
483 sub includemhproblem {
484   my ($gullet,$keyval,$arg2) = @_ ;
485   my $repo_path;
486   if ($keyval) {
487     $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
488   if (! $repo_path) {
489     $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }
490   else {
491     $keyval->setValue('mhrepos',undef); }
492   my $mathhub_base = ToString(Digest('\MathHub{'}));

```

```

493 my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);
494 return Invocation(T_CS('\includeproblem'), $keyval, T_OTHER($finalpath)); }#$
495 DefKeyVal('inclprob','mhrepos','Semiverbatim');
496 DefMacro('\includemhproblem OptionalKeyVals:inclprob {}', \&includemhproblem);
497 </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.

```

498 <*hwexam>
499 \ProvidesPackage{hwexam-mh}[2015/11/04 v1.0 MathHub support for the sTeX hwexam package]
500 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{hwexam}}
501 \ProcessOptions
502 \RequirePackage{hwexam}
503 \RequirePackage{mathhub}
504 </hwexam>
505 <*hwexam.ltxml>
506 DeclareOption(undef,sub{PassOptions('hwexam','sty',ToString(Digest(T_CS('\CurrentOption')))); }
507 ProcessOptions();
508 RequirePackage('hwexam');
509 RequirePackage('mathhub');
510 </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`.

```

511 <*package>
512 \newcommand\includemhassignment[2][]{\metasetkeys{inclassig}{#1}%
513 \edef\mh@@repos{\mh@currentrepos}%
514 \ifx\inclassig@mhrepos\empty\else\mhcurrentrepos\inclassig@mhrepos\fi%
515 \includeassignment[#1]{\MathHub{\mh@currentrepos/source/#2}}%
516 \mhcurrentrepos\mh@@repos\clear@inclassig@keys}
517 </package>
518 <*ltxml>
519 sub includemhassignment {
520 my ($gullet,$keyval,$arg2) = @_ ;
521 my $repo_path;
522 if ($keyval) {
523     $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
524 if (! $repo_path) {
525     $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }
526 else {
527     $keyval->setValue('mhrepos',undef); }
528 my $mathhub_base = ToString(Digest('\MathHub{'}));
529 my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);
530 return Invocation(T_CS('\includeassignment'), $keyval, T_OTHER($finalpath)); }#$
531 DefKeyVal('inclprob','mhrepos','Semiverbatim');

```

```

532 DefMacro('\includemhassignment OptionalKeyVals:inclprob {}', \&includemhassignment);
533 </ltxml>

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

```

4.9 Finale

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

```

556 <*ltxml>
557 1;
558 </ltxml>

```