

MathHub Support for \LaTeX^*

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

November 5, 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 [Hor+11]. 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 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 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].

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 `\beginxml` and `\endxml`). 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 \beginxml | modules.xml | omtex.xml | multiling.xml | mikosides.xml | problem.xml | hwexam.xml
2 \package \LaTeXXML::Package::Pool;
3 \use strict;
4 \use \LaTeXXML::Package;
5 \endxml | modules.xml | omtex.xml | multiling.xml | mikosides.xml | problem.xml | hwexam.xml
6 \package\ProvidesPackage{mathhub}[2015/11/04 v1.0 sTeX Support for MathHub.info]

7 \beginpackage
8 \DeclareOption*{}
9 \ProcessOptions
10 \endpackage
11 \beginxml
12 \DeclareOption(undef,sub {});
13 \ProcessOptions();
14 \endxml

```

Then we need to set up the packages by requiring the `metakeys` package [Koh15] to be loaded (in the right version).

```

15 \beginpackage
16 \RequirePackage{keyval}
17 \endpackage
18 \beginxml
19 \RequirePackage('keyval');
20 \endxml

```

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

```

```

32 </package>
33 <*ltxml>
34 DefMacro('mhcurrentrepos{','\@mhcurrentrepos{#1}');
35 DefMacro('mhcurrentrepos{','\def\mh@currentrepos{#1}\@mhcurrentrepos{#1}');
36 DefConstructor('mhcurrentrepos{','',
37   afterDigest => sub{ AssignValue('current_repos',ToString($_[1]->getArg(1)),'global'); } );
38 </ltxml>#<
\libinput the \libinput macro inputs from the lib directory of the MathHub repository
or the meta-inf/lib repos of the group.
39 <ltxml>RaxTeX('
40 <*package | ltxml>
41 \def\modules@@first#1/#2;{#1}
42 \newcommand\libinput[1]{\def\@libfile{\MathHub{\mh@currentrepos/lib/#1}}%
43 \IfFileExists{\@libfile}{\input\@libfile}%
44 {\edef\@group{\expandafter\modules@@first\mh@currentrepos;}
45 \edef\@infile{\MathHub{\@group/meta-inf/lib/#1}}
46 \IfFileExists{\@infile}{\input{\@infile}}%
47 {\PackageError{modules}
48   {Library file missing, cannot input #1\MessageBreak%
49     Both \@libfile.tex\MessageBreak and \@infile.tex\MessageBreak do not exit}%
50   {Check whether the file name is correct}}}%
51 </package | ltxml>
52 <ltxml>');

```

4.2 modules-mh: MH Variants for Modules

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

```

53 <*modules>
54 \ProvidesPackage{modules-mh}[2015/11/04 v1.0 MathHub support for the sTeX modules package]
55 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{modules}}
56 \ProcessOptions
57 \RequirePackage{modules}
58 \RequirePackage{mathhub}
59 </modules>
60 <*modules.ltxml>
61 DeclareOption(undef,sub{PassOptions('modules','sty',ToString(Digest(T_CS('CurrentOption')))));
62 ProcessOptions();
63 RequirePackage('modules');
64 RequirePackage('mathhub');
65 </modules.ltxml>
\importmhmodule The \importmhmodule[<key=value list>]{module} saves the current value of
\mh@currentrepos in a local macro \mh@@repos, resets \mh@currentrepos to
the new value if one is given in the optional argument, and after importing resets
\mh@currentrepos to the old value in \mh@@repos. We do all the \ifx compar-
ison with an \expandafter, since the values may be passed on from other key
bindings. Parameters will be passed to \importmodule.

```

```

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

```



```

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

```

and now the analogs

\usemhmodule

```

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

```

\mhinputref

```

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

```

```

160 </modules | modules.ltxml>
161 <modules.ltxml>');

```

`\mhinput`

```

162 \let\mhinput\mhinputref%

```

4.3 omtex-mh: MH Variants for OMText

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

```

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

```

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

```

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

```

```

199 return Invocation(T_CS('@includegraphicx'), $keyval, T_OTHER($finalpath)); }#$
200 DefKeyVal('Gin', 'mhrepos', 'Semiverbatim');
201 DefMacro('\mhgraphics OptionalKeyVals:Gin {}', \&mhgraphics);
202 DefMacro('\mhcgraphics []{}', '\begin{center}\mhgraphics[#1]{#2}\end{center}');
203 DefMacro('\mhbgraphics []{}', '\fbox{\mhgraphics[#1]{#2}}');
204 </omtext.ltxml>

```

4.4 smultiling-mh: MH Variants for Multilinguality

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

```

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

```

`mhmodnl:`

```

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

```

`mhmodnl` The `mhmodnl` environment is just a layer over the module environment and the `\importmhmodule` macro with the keys and language suitably adapted.

```

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

```

```

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

```

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

```

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

```

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

EdN:4

with the keys and language suitably adapted.⁴

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

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

`importmhmodulevia`

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

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

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

```

mhview the MathHub version

```

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

```

```

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

```

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

```

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

```

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

```

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

```

```

414 DefMacroI('end{mhviewsketch}',undef,'end{viewsketchenv}');
415 </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.

```

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

```

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

```

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

```

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

```



```

449 \RequirePackage{mathhub}
450 \end{problem}
451 \problem{problem}
452 \DeclareOption{undef}{\PassOptionsToPackage{problem}{sty}{\ToString(Digest(T_CS('CurrentOption')))});
453 \ProcessOptions{};
454 \RequirePackage{problem};
455 \RequirePackage{mathhub};
456 \end{problem}

```

`\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`.

```

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

```

4.8 hwexam-mh: Support for Assignments

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

```

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

```

```

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

```

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

```

`\inputmhassignment` analogous

```

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

```

```

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

```

4.9 Finale

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

```

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

```

References

- [Hor+11] Fulya Horozal et al. “Combining Source, Content, Presentation, Narration, and Relational Representation”. In: *Intelligent Computer Mathematics*. Ed. by James Davenport et al. LNAI 6824. Springer Verlag, 2011, pp. 212–227. ISBN: 978-3-642-22672-4. URL: http://kwarc.info/frabe/Research/HIJKR_dimensions_11.pdf.
- [Koh15] Michael Kohlhasse. *metakeys.sty: A generic framework for extensible Metadata in L^AT_EX*. Tech. rep. Comprehensive T_EX Archive Network (CTAN), 2015. URL: <http://www.ctan.org/tex-archive/macros/latex/contrib/stex/metakeys/metakeys.pdf>.
- [sTeX] *KWARC/sTeX*. URL: <https://svn.kwarc.info/repos/stex> (visited on 05/15/2015).