

hwexam.sty/cls: An Infrastructure for formatting Assignments and Exams*

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

July 6, 2015

Abstract

The `hwexam` package and class allows individual course assignment sheets and compound assignment documents using problem files marked up with the `problem` package.

Contents

1	Introduction	2
2	The User Interface	2
2.1	Package and Class Options	2
2.2	Assignments	2
2.3	Typesetting Exams	2
2.4	Including Assignments	3
2.5	Support for MathHub	4
3	Limitations	4
4	Implementation: The hwexam Class	6
4.1	Class Options	6
5	Implementation: The hwexam Package	7
5.1	Package Options	7
5.2	Assignments	8
5.3	Including Assignments	11
5.4	Typesetting Exams	12
5.5	Support for MathHub	14
5.6	Leftovers	15

*Version v1.0 (last revised 2013/12/12)

1 Introduction

The `hwexam` package and class supplies an infrastructure that allows to format nice-looking assignment sheets by simply including problems from problem files marked up with the `problem` package [Kohlhase:problem:ctan]. It is designed to be compatible with `problems.sty`, and inherits some of the functionality.

2 The User Interface

2.1 Package and Class Options

The `hwexam` package and class take the options `solutions`, `notes`, `hints`, `pts`, `min`, and `boxed` that are just passed on to the `problems` package (cf. its documentation for a description of the intended behavior).

`showmeta` If the `showmeta` option is set, then the metadata keys are shown (see [Kohlhase:metakeys:ctan] for details and customization options).

The `hwexam` class additionally accepts the options `report`, `book`, `chapter`, `part`, and `showignores`, of the `omdoc` package [Kohlhase:smomdl:ctan] on which it is based and passes them on to that. For the `extrefs` option see [Kohlhase:sref:ctan].

2.2 Assignments

`assignment` This package supplies the `assignment` environment that groups problems into assignment sheets. It takes an optional `KeyVal` argument with the keys `number` (for the assignment number; if none is given, 1 is assumed as the default or — in multi-assignment documents — the ordinal of the `assignment` environment), `title` (for the assignment title; this is referenced in the title of the assignment sheet), `type` (for the assignment type; e.g. “quiz”, or “homework”), `given` (for the date the assignment was given), and `due` (for the date the assignment is due).

2.3 Typesetting Exams

`multiple` Furthermore, the `hwexam` package takes the option `multiple` that allows to combine multiple assignment sheets into a compound document (the assignment sheets are treated as section, there is a table of contents, etc.).

`test` Finally, there is the option `test` that modifies the behavior to facilitate formatting tests. Only in `test` mode, the macros `\testspace`, `\testnewpage`, and `\testemptypage` have an effect: they generate space for the students to solve the given problems. Thus they can be left in the \LaTeX source.

`\testspace` `\testspace` takes an argument that expands to a dimension, and leaves vertical space accordingly. `\testnewpage` makes a new page in `test` mode, and `\testemptypage` generates an empty page with the cautionary message that this page was intentionally left empty.

`testheading` Finally, the `\testheading` takes an optional keyword argument where the keys `duration` specifies a string that specifies the duration of the test, `min` specifies the

`reqpts` equivalent in number of minutes, and `reqpts` the points that are required for a perfect grade.

```
\title{320101 General Computer Science (Fall 2010)}
\begin{testheading}[duration=one hour,min=60,reqpts=27]
  Good luck to all students!
\end{testheading}
```

formats to

Name:

Matriculation Number:

320101 General Computer Science (Fall 2010)

July 6, 2015

You have one hour(sharp) for the test;

Write the solutions to the sheet.

The estimated time for solving this exam is 58 minutes, leaving you 2 minutes for revising your exam.

You can reach 30 points if you solve all problems. You will only need 27 points for a perfect score, i.e. 3 points are bonus points.

Different problems test different skills and knowledge, so do not get stuck on one problem.

	To be used for grading, do not write here								
prob.	1.1	2.1	2.2	2.3	3.1	3.2	3.3	Sum	grade
total	4	4	6	6	4	4	2	30	
reached									

good luck

Example 1: A generated test heading.

2.4 Including Assignments

`\includeassignment` The `\includeassignment` macro can be used to include an assignment from another file. It takes an optional `KeyVal` argument and a second argument which is a path to the file containing the problem (the macro assumes that there is only one `assignment` environment in the included file). The keys `number`, `title`, `type`, `given`, and `due` are just as for the `assignment` environment and (if given) overwrite the ones specified in the `assignment` environment in the included file.

2.5 Support for MathHub

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 `modules` macros.

`\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}
```

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

```
\includemhassignment{baz/foobar}
```

Of course, neither \LaTeX nor $\text{\LaTeX}ML$ know about the repositories when they are called from a file system, so we can use the `\mhcurrentrepos` macro from the `modules` package 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.

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.

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 \LaTeX TRAC [[sTeX:online](http://www.tex.ac.uk)].

1. none reported yet.

4 Implementation: The hwexam Class

The functionality is spread over the `hwexam` class and package. The class provides the `document` environment and pre-loads some convenience packages, whereas the package provides the concrete functionality.

`hwexam.dtx` generates four files: `hwexam.cls` (all the code between `<*cls>` and `</cls>`), `hwexam.sty` (between `<*package>` and `</package>`) and their L^AT_EXML bindings (between `<*ltxml.cls>` and `</ltxml.cls>` and `<*ltxml.sty>` and `</ltxml.sty>` respectively). We keep the corresponding code fragments together, since the documentation applies to both of them and to prevent them from getting out of sync.

4.1 Class Options

To initialize the `hwexam` class, we declare and process the necessary options by passing them to the respective packages and classes they come from.

```
1 <*cls>
2 \DeclareOption{test}{\PassOptionsToPackage{\CurrentOption}{hwexam}}
3 \DeclareOption{multiple}{\PassOptionsToPackage{\CurrentOption}{hwexam}}
4 \DeclareOption{showmeta}{\PassOptionsToPackage{\CurrentOption}{metakeys}}
5 \DeclareOption{extrefs}{\PassOptionsToPackage{\CurrentOption}{sref}}
6 \DeclareOption{notes}{\PassOptionsToPackage{\CurrentOption}{problem}}
7 \DeclareOption{hints}{\PassOptionsToPackage{\CurrentOption}{problem}}
8 \DeclareOption{solutions}{\PassOptionsToPackage{\CurrentOption}{problem}}
9 \DeclareOption{pts}{\PassOptionsToPackage{\CurrentOption}{problem}}
10 \DeclareOption{min}{\PassOptionsToPackage{\CurrentOption}{problem}}
11 \DeclareOption{boxed}{\PassOptionsToPackage{\CurrentOption}{problem}}
12 \DeclareOption{extract}{\PassOptionsToPackage{\CurrentOption}{problem}}
13 \DeclareOption*{\PassOptionsToClass{\CurrentOption}{omdoc}}
14 \ProcessOptions
15 </cls>
16 <*ltxml.cls>
17 # -*- CPERL -*-
18 package LaTeXML::Package::Pool;
19 use strict;
20 use LaTeXML::Package;
21 use LaTeXML::Util::Pathname;
22 use Cwd qw(cwd abs_path);
23 DeclareOption('test',sub {PassOptions('hwexam','sty',ToString(Digest(T_CS('\CurrentOption'))));});
24 DeclareOption('multiple',sub {PassOptions('hwexam','sty',ToString(Digest(T_CS('\CurrentOption'))));});
25 DeclareOption('showmeta',sub {PassOptions('metakeys','sty',ToString(Digest(T_CS('\CurrentOption'))));});
26 DeclareOption('extrefs',sub {PassOptions('sref','sty',ToString(Digest(T_CS('\CurrentOption'))));});
27 DeclareOption('notes',sub {PassOptions('problem','sty',ToString(Digest(T_CS('\CurrentOption'))));});
28 DeclareOption('hints',sub {PassOptions('problem','sty',ToString(Digest(T_CS('\CurrentOption'))));});
29 DeclareOption('solutions',sub {PassOptions('problem','sty',ToString(Digest(T_CS('\CurrentOption'))));});
30 DeclareOption('pts',sub {PassOptions('problem','sty',ToString(Digest(T_CS('\CurrentOption'))));});
31 DeclareOption('min',sub {PassOptions('problem','sty',ToString(Digest(T_CS('\CurrentOption'))));});
32 DeclareOption('boxed',sub {PassOptions('problem','sty',ToString(Digest(T_CS('\CurrentOption'))));});
33 DeclareOption('extract',sub {PassOptions('problem','sty',ToString(Digest(T_CS('\CurrentOption'))));});
```

```

34 DeclareOption(undef,sub {PassOptions('omdoc','cls',ToString(Digest(T_CS('\CurrentOption')))); }
35 ProcessOptions();
36 \ltxml.cls)

```

We load `omdoc.cls`, and the desired packages. For the L^AT_EXML bindings, we make sure the right packages are loaded.

```

37 \*cls)
38 \LoadClass{omdoc}
39 \RequirePackage{stex}
40 \RequirePackage{hwexam}
41 \RequirePackage{graphicx}
42 \RequirePackage{a4wide}
43 \RequirePackage{amssymb}
44 \RequirePackage{amstext}
45 \RequirePackage{amsmath}
46 \*cls)
47 \ltxml.cls)
48 LoadClass('omdoc');
49 RequirePackage('stex');
50 RequirePackage('hwexam');
51 RequirePackage('graphicx');
52 RequirePackage('amssymb');
53 RequirePackage('amstext');
54 RequirePackage('amsmath');
55 \ltxml.cls)

```

5 Implementation: The hwexam Package

5.1 Package Options

The first step is to declare (a few) package options that handle whether certain information is printed or not. Some come with their own conditionals that are set by the options, the rest is just passed on to the `problems` package.

```

56 \*package)
57 \DeclareOption{showmeta}{\PassOptionsToPackage{\CurrentOption}{metakeys}}
58 \newif\iftest\testfalse
59 \newif\ifsolutions\solutionsfalse
60 \DeclareOption{test}{\testtrue\solutionsfalse}
61 \newif\ifmultiple\multiplefalse
62 \DeclareOption{multiple}{\multipletrue}
63 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{problem}}
64 \ProcessOptions
65 \*package)

```

Then we make sure that the necessary packages are loaded (in the right versions).

```

66 \*package)
67 \RequirePackage{keyval}[1997/11/10]
68 \RequirePackage{problem}
69 \*package)

```

Here comes the equivalent header information for L^AT_EXML, we also initialize the package inclusions. Since L^AT_EXML does not handle options yet, we have nothing to do.

```

70 <*txml>
71 # -*- CPERL -*-
72 package LaTeXML::Package::Pool;
73 use strict;
74 use LaTeXML::Package;
75 DeclareOption('test', '');
76 DeclareOption('multiple', '');
77 DeclareOption('showmeta', sub {PassOptions('metakeys','sty',ToString(Digest(T_CS('\CurrentOption'))));});
78 DeclareOption(undef, sub {PassOptions('problem','sty',ToString(Digest(T_CS('\CurrentOption'))));});
79 ProcessOptions();
80 RequirePackage('problem');
81 </txml>

```

Then we register the namespace of the requirements ontology

```

82 <*txml>
83 RegisterNamespace('assig'=>"http://omdoc.org/ontology/assignments#");
84 RegisterDocumentNamespace('assig'=>"http://omdoc.org/ontology/assignments#");
85 </txml>

```

5.2 Assignments

We will prepare the keyval support for the `assignment` environment.

```

86 <*package>
87 \srefaddidkey{assig}
88 \addmetakey{assig}{number}
89 \addmetakey*{assig}{title}
90 \addmetakey{assig}{type}
91 \addmetakey{assig}{given}
92 \addmetakey{assig}{due}

```

The next three macros are intermediate functions that handle the case gracefully, where the respective token registers are undefined.

The `\given@due` macro prints information about the given and due status of the assignment. Its arguments specify the brackets.

```

93 \newcommand\given@due[2]{%
94 \ifx \inclassig@given\@empty
95 \ifx \assig@given\@empty
96 \ifx \inclassig@due\@empty
97 \ifx \assig@due\@empty% all empty do nothing
98 \else #1%
99 \fi
100 \else #1%
101 \fi
102 \else #1%
103 \fi
104 \else #1%

```



```

105 \fi
106 \ifx\inclassig@given\@empty
107 \ifx\assig@given\@empty% do nothing
108 \else Given \assig@given%
109 \fi
110 \else Given \inclassig@given%
111 \fi
112 \ifx \inclassig@due\@empty
113 \ifx \assig@due\@empty% do nothing
114 \else
115 \ifx \inclassig@given\@empty
116 \ifx \assig@given\@empty% do nothing
117 \else ,~%
118 \fi
119 \else ,~%
120 \fi
121 \fi
122 \else
123 \ifx \inclassig@given\@empty
124 \ifx \assig@given\@empty% do nothing
125 \else ,~%
126 \fi
127 \else ,~%
128 \fi
129 \fi
130 \ifx \inclassig@due\@empty
131 \ifx \assig@due\@empty% do nothing
132 \else Due \assig@due%
133 \fi
134 \else Due \inclassig@due%
135 \fi
136 \ifx \inclassig@given\@empty
137 \ifx \assig@given\@empty
138 \ifx \inclassig@due\@empty
139 \ifx \assig@due\@empty% all empty do nothing
140 \else #2%
141 \fi
142 \else #2%
143 \fi
144 \else #2%
145 \fi
146 \else #2%
147 \fi
148 }

```

assignment@title This macro prints the title of an assignment, the local title is overwritten, if there is one from the `\includeassignment`. `\assignment@title` takes three arguments the first is the fallback when no title is given at all, the second and third go around the title, if one is given.

```

149 \newcommand\assignment@title[3]
150 {\ifx\inclassig@title\empty% if there is no outside title
151 \ifx\assig@title\empty{#1}\else{#2\assig@title{#3}}\fi
152 \else{#2}\inclassig@title{#3}\fi}% else show the outside title

```

With them, we can define the central `assignment` environment. This has two forms (separated by `\ifmultiple`) in one we make a title block for an assignment sheet, and in the other we make a section heading and add it to the table of contents.

`assignment@titleblock` This macro prints the title block of a section. If the `multiple` package option is given we make a section heading out of this, and if not, a title block.

```

153 \ifmultiple
154 \newcommand\assignment@titleblock{%
155 \section*{\protect\document@hwexamtype~\arabic{section}%
156 \assignment@title{\;(\;)\;}\given@due{\;}}%
157 \addcontentsline{toc}{section}%
158 {\document@hwexamtype~\arabic{section}}:~%
159 \string\importmodules{\imported@modules}\assig@title}%
160 \setcounter{problem}{0}}

```

Now to the single assignment case, where we make a title block. Note that as problems are numbered by section, we also set the section counter.

```

161 \else% single
162 \newcommand\assignment@titleblock{%
163 \begin{center}\bf
164 \Large@title\strut\
165 \document@hwexamtype~\arabic{section}\assignment@title{\;:\;}\{\}\%
166 \large\given@due{--\;}\{\;--}
167 \end{center}}
168 \fi% ifmultiple

```

`assignment`

```

169 \newenvironment{assignment}[1][\metasetkeys{assig}{#1}\sref@target%
170 \edef\@@num{\ifx\inclassig@number\empty%
171 \ifx\assig@number\empty\else\assig@number\fi%
172 \else\inclassig@number\fi}%
173 \ifx\@@num\empty\stepcounter{section}\else\setcounter{section}{\@@num}\fi%
174 \sref@label{id{Assignment \thesection}}%
175 \assignment@titleblock%
176 \def\currentsectionlevel{assignment\hspace}%
177 \def\Currentsectionlevel{Assignment\hspace}%
178 \ignorespaces{}
179 \</package>

180 \<?xml>
181 DefEnvironment('assignment' OptionalKeyVals:assig',
182 " <omdoc:omgroup ?&GetKeyVal(#1,'id')(xml:id='&GetKeyVal(#1,'id')')() "
183 . "assig:dummy='for the namespace'"
184 . "<omdoc:metadata>"

```

```

185 .      "<dc:title>"
186 .      "Assignment ?&GetKeyVal(#1,'num')(&GetKeyVal(#1,'num').)()"
187 .      "?&GetKeyVal(#1,'title')(&GetKeyVal(#1,'title')))"
188 .      "</dc:title>"
189 .      "?&GetKeyVal(#1,'given')(<omdoc:meta property='assig:given'>&GetKeyVal(#1,'given')</omdo
190 .      "?&GetKeyVal(#1,'due')(<omdoc:meta property='assig:due'>&GetKeyVal(#1,'due')</omdoc:meta
191 .      "?&GetKeyVal(#1,'pts')(<omdoc:meta property='assig:pts'>&GetKeyVal(#1,'pts')</omdoc:meta
192 .      "</omdoc:metadata>"
193 .      "#body"
194 .      "</omdoc:omgroup>\n"#,
195 #      afterDigest=> sub {
196 #          my ($stomach, $kv) = @_;
197 #          my $kvi = LookupValue('inclassig');
198 #          my @keys = qw(id num title pts given due);
199 #          my @vals = $kvi && map($kvi->getValue($_), @keys);
200 #          foreach my $i(0..$#vals) {
201 #              $kv->setValue($keys[$i],$vals[$i]) if $vals[$i];
202 #          }
203 #      };#$
204 </ltxml>

205 <*package>
206 \newcommand\assig@default@type{Assignment}
207 \addmetakey[\assig@default@type]{document}{hwexamtype}
208 </package>

```

5.3 Including Assignments

`\in*assignment` This macro is essentially a glorified `\include` statement, it just sets some internal macros first that overwrite the local points. Importantly, it resets the `inclassig` keys after the input.

```

209 <*package>
210 \addmetakey{inclassig}{number}
211 \addmetakey*{inclassig}{title}
212 \addmetakey{inclassig}{type}
213 \addmetakey{inclassig}{given}
214 \addmetakey{inclassig}{due}
215 \addmetakey{inclassig}{mhrepos}
216 \clear@inclassig@keys%initially
217 \newcommand\includeassignment[2][\metasetkeys{inclassig}{#1}%
218 \include{#2}\clear@inclassig@keys}
219 \newcommand\inputassignment[2][\metasetkeys{inclassig}{#1}%
220 \input{#2}\clear@inclassig@keys}
221 </package>
222 <*ltxml>
223 DefMacro('includeassignment [] {}', sub {
224     my ($stomach, $arg1, $arg2) = @_;
225     AssignValue('inclassig',$arg1) if $arg1;
226     (Invocation(T_CS('input'),$arg2)->unlist);

```

```

227 });
228 DefMacro('inputassignment [] {}','includeassignment[#1]{#2}');
229 
```

5.4 Typesetting Exams

`\quizheading`

```

230 (*package)
231 \addmetakey{quizheading}{tas}
232 \newcommand\quizheading[1]{\def\tas{#1}%
233 \large\noindent NAME: \hspace{8cm} MAILBOX: \hspace{2cm}%
234 \ifx\tas\empty\else%
235 \noindent TA: \for\@I:=\tas\do{\Large$\Box$\@I\hspace*{1em}}\hspace{2cm}\fi}

```

`\testheading`

```

236 \addmetakey{testheading}{min}
237 \addmetakey{testheading}{duration}
238 \addmetakey{testheading}{reqpts}
239 \newenvironment{testheading}[1][\metasetkeys{testheading}{#1}
240 {\noindent\large{Name: \hfill Matriculation Number: \hspace*{2cm}\strut\hspace{1cm}}
241 \begin{center}\Large\textbf{\@title}\hspace{1cm}\large\@date\hspace{3cm}\end{center}
242 {\textbf{You have
243 \ifx\test@heading@duration\empty\testheading@min minutes\else\testheading@duration\fi
244 (sharp) for the test}};\hspace{1cm}Write the solutions to the sheet.}\par\noindent
245
246 \newcount\check@time\check@time=\testheading@min
247 \advance\check@time by -\theassignment@totalmin
248 The estimated time for solving this exam is {\theassignment@totalmin} minutes,
249 leaving you {\the\check@time} minutes for revising your exam.
250
251 \newcount\bonus@pts\bonus@pts=\theassignment@totalpts
252 \advance\bonus@pts by -\testheading@reqpts
253 You can reach {\theassignment@totalpts} points if you solve all problems. You will only need
254 {\testheading@reqpts} points for a perfect score, i.e. \hspace{1cm}{\the\bonus@pts} points are
255 bonus points. \hspace{1cm}
256 \begin{center}
257 {\Large\em
258 % You have ample time, so take it slow and avoid rushing to mistakes!\hspace{2cm}
259 Different problems test different skills and knowledge, so do not get stuck on
260 one problem.}\hspace{1cm}\par\correction@table \hspace{3cm}
261 \end{center}}
262 {\newpage}
263 
```

`\testspace`

```

267 (*package)

```

```

268 \newcommand\testspace[1]{\iftest\vspace*{#1}\fi}
269 \end{package}
270 \end{xml}
271 DefConstructor('\testspace','');
272 \end{xml}

\testnewpage
273 \end{package}
274 \newcommand\testnewpage{\iftest\newpage\fi}
275 \end{package}
276 \end{xml}
277 DefConstructor('\testnewpage','');
278 \end{xml}

\testemptypage
279 \end{package}
280 \newcommand\testemptypage[1][\iftest\begin{center}This page was intentionally left
281     blank for extra space\end{center}\vfill\ject\else\fi}
282 \end{package}
283 \end{xml}
284 DefConstructor('\testemptypage','');
285 \end{xml}

\@problem This macro acts on a problem's record in the *.aux file. Here we redefine it to
generate the correction table.
286 \end{package}
287 \renewcommand\@problem[3]{\stepcounter{assignment@probs}
288 \def\@pts{#2}\ifx\@pts\@empty\else\addtocounter{assignment@totalpts}{#2}\fi
289 \def\@min{#3}\ifx\@min\@empty\else\addtocounter{assignment@totalmin}{#3}\fi
290 \xdef\correction@probs{\correction@probs & #1}%
291 \xdef\correction@pts{\correction@pts & #2}
292 \xdef\correction@reached{\correction@reached &}}
293 \end{package}

\correction@table This macro generates the correction table
294 \end{package}
295 \newcounter{assignment@probs}
296 \newcounter{assignment@totalpts}
297 \newcounter{assignment@totalmin}
298 \newcommand\correction@probs{prob.}%
299 \newcommand\correction@pts{total}%
300 \newcommand\correction@reached{reached}%
301 \stepcounter{assignment@probs}
302 \newcommand\correction@table{\begin{tabular}{|l|*{\theassignment@probs}{c|}|l|}\hline%
303 &\multicolumn{\theassignment@probs}{c|}|%
304 {\footnotesize To be used for grading, do not write here} &\\\hline
305 \correction@probs & Sum & grade\\\hline
306 \correction@pts & \theassignment@totalpts & \\\hline
307 \correction@reached & & \[.7cm]\hline

```

```

308 \end{tabular}}
309 \end{package}

```

5.5 Support for MathHub

`\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`.

```

310 \newcommand\includemhassignment[2] [] {\metasetkeys{inclassig}{#1}%
311 \edef\mh@@repos{\mh@currentrepos}%
312 \ifx\inclassig@mhrepos\empty\else\mhcurrentrepos\inclassig@mhrepos\fi%
313 \includeassignment[#1]{\MathHub{\mh@currentrepos/source/#2}}%
314 \mhcurrentrepos\mh@@repos\clear@inclassig@keys}
315 \end{package}
316 \end{package}
317 \end{package}
318 sub includemhassignment {
319   my ($gullet,$keyval,$arg2) = @_;
320   my $repo_path;
321   if ($keyval) {
322     $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
323   if (! $repo_path) {
324     $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }
325   else {
326     $keyval->setValue('mhrepos',undef); }
327   my $mathhub_base = ToString(Digest('\MathHub{'}));
328   my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);
329   return Invocation(T_CS('includeassignment'), $keyval, T_OTHER($finalpath)); }#
330 DefKeyVal('inclprob','mhrepos','Semiverbatim');
331 DefMacro('\includemhassignment OptionalKeyVals:inclprob {}', \&includemhassignment);
332 \end{package}

```

`\inputmhassignment` analogous

```

333 \newcommand\inputmhassignment[2] [] {\metasetkeys{inclassig}{#1}%
334 \edef\mh@@repos{\mh@currentrepos}%
335 \ifx\inclassig@mhrepos\empty\else\mhcurrentrepos\inclassig@mhrepos\fi%
336 \inputassignment[#1]{\MathHub{\mh@currentrepos/source/#2}}%
337 \mhcurrentrepos\mh@@repos\clear@inclassig@keys}
338 \end{package}
339 \end{package}
340 \end{package}
341 sub inputmhassignment {
342   my ($gullet,$keyval,$arg2) = @_;
343   my $repo_path;
344   if ($keyval) {
345     $repo_path = ToString(GetKeyVal($keyval,'mhrepos')); }
346   if (! $repo_path) {
347     $repo_path = ToString(Digest(T_CS('\mh@currentrepos'))); }

```

```

348   else {
349     $keyval->setValue('mhrepos',undef); }
350   my $mathhub_base = ToString(Digest('\MathHub{'}));
351   my $finalpath = $mathhub_base.$repo_path.'/source/'.ToString($arg2);
352   return Invocation(T_CS('\inputassignment'), $keyval, T_OTHER($finalpath)); }#$
353 DefMacro('\inputmhassignment OptionalKeyVals:inclprob {}', \&inputmhassignment);
354 </ltxml>

```

5.6 Leftovers

at some point, we may want to reactivate the logos font, then we use

```

here we define the logos that characterize the assignment
\font\bierfont=../assignments/bierglas
\font\denkerfont=../assignments/denker
\font\uhrfont=../assignments/uhr
\font\warnschildfont=../assignments/achtung

\newcommand\bierglas{{\bierfont\char65}}
\newcommand\denker{{\denkerfont\char65}}
\newcommand\uhr{{\uhrfont\char65}}
\newcommand\warnschild{{\warnschildfont\char 65}}
\newcommand\hardA{\warnschild}
\newcommand\longA{\uhr}
\newcommand\thinkA{\denker}
\newcommand\discussA{\bierglas}

```

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

```

355 <ltxml>1;

```